TECHNICAL DOCUMENT 3033
August 1998

Advanced Propagation Model (APM) Computer Software Configuration Item (CSCI) Documents

D. R. Sailors
A. E. Barrios
W. L. Patterson
H. V. Hitney

SPAWAR
Systems Center
San Diego

Space and Naval Warfare Systems Center
San Diego, CA 92152–5001

SPACE AND NAVAL WARFARE SYSTEMS CENTER
San Diego, California 92152–5001

H. A. Williams, CAPT, USN
Commanding Officer

R. C. Kolb
Executive Director

ADMINISTRATIVE INFORMATION

The work detailed in this report was performed for Space and Naval Warfare Systems Command, PMW–185 by the Tropospheric Branch, Code D883, of Space and Naval Warfare (SPAWAR) Systems Center, San Diego. Funding for this project was provided under program element 0603207N.

Released by
R. A. Paulus, Head
Tropospheric Branch

Under authority of
J. H. Richter, Head
Propagation Division

SB

SOFTWARE REQUIREMENTS SPECIFICATION

FOR THE

ADVANCED PROPAGATION MODEL CSCI

(Version 1.0)

August 1998

Prepared for:

Space and Naval Warfare Systems Command (PMW-185)

San Diego, CA

Prepared by:

Space and Naval Warfare Systems Center San Diego

Tropospheric Branch (Code D883)

49170 Propagation Path

San Diego, CA 92152-7385

CONTENTS

Figures

Tables

1. SCOPE

1.1 IDENTIFICATION

The Advanced Propagation Model (APM) Version 1.0 computer software configuration item (CSCI) calculates range-dependent electromagnetic (EM) system propagation loss within a heterogeneous atmospheric medium over variable terrain, where the radio-frequency index of refraction is allowed to vary both vertically and horizontally, also accounting for terrain effects along the path of propagation.

1.2 SYSTEM OVERVIEW

The APM CSCI model will calculate propagation loss values as EM energy propagates through a laterally heterogeneous atmospheric medium where the index of refraction is allowed to vary both vertically and horizontally, also accounting for terrain effects along the path of propagation. Numerous Tactical Environmental Support System-Next Century (TESS-NC) applications require EM-system propagation loss values. The required APM model described by this document may be applied to two such TESS-NC applications, one of which displays propagation loss on a range versus height scale (commonly referred to as a coverage diagram) and one which displays propagation loss on a propagation loss versus range/height scale (commonly referred to as a loss diagram).

1.3 DOCUMENT OVERVIEW

This document specifies the functional requirements that are to be met by the APM CSCI. A discussion of the input software requirements is presented together with a general description of the internal structure of the APM CSCI as it relates to the CSCI's capability.

2. REFERENCED DOCUMENTS

(a) Bergland, G. D. 1969. "A Radix-eight Fast Fourier Transform Subroutine for Real-valued Series," *IEEE Trans. Audio and Electro-acoust.*, vol. AU-17, pp. 138-144.

(b) Cooley, J. W., P. A. W. Lewis and P. D. Welsh. 1970. "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculation of Sine, Cosine and Laplace Transforms," *J. Sound Vib.*, vol. 12, pp. 315-337.

(c) Tappert, F. D. 1977. "The Parabolic Approximation Method," Wave Propagation and Underwater Acoustics, pp. 224–285, J. B. Keller and J. S. Papadakis, Eds., Springer-Verlag, New York, NY.

(d) International Radio Consulting Committee (CCIR) XVth Plenary Assembly Dubrovnik, 1986. "Propagation in Non-Ionized Media," *Recommendations and Reports of the CCIR, 1986*, vol. V, International Telecommunications Union, Geneva, Switzerland.

(e) Commander-In-Chief, Pacific Fleet Meteorological Requirement (PAC MET) 87-04. 1997. "Range Dependent Electromagnetic Propagation Models."

(f) Dockery, G. D. 1988. "Modeling Electromagnetic Wave Propagation in the Troposphere Using the Parabolic Equation", *IEEE Trans. Antennas Propagat.*, vol. 36, no. 10, pp. 1464–1470 (Oct).

(g) Naval Oceanographic Office. 1990. "Software Documentation Standards and Coding Requirements for Environmental System Product Development," Apr.

(h) Kuttler, J. R. and G. D. Dockery. 1991. "Theoretical Description of the Parabolic Approximation/Fourier Split-Step Method of Representing Electromagnetic Propagation in the Troposphere," Radio Sci., Vol. 26, pp. 381–393.

(i) American National Standards Institute (ANSI). 1992. "Program Language - Fortran - Extended."

(j) Patterson, W.L. and H. V. Hitney. 1992. "Radio Physical Optics CSCI Software Documents." NraD TD 2403 (Dec), Naval Command, Control and Ocean Surveillance Center RDT&E Division, San Diego.

(k) Barrios, A. E. 1993. "Terrain and Refractivity Effects on Non-Optical Paths," AGARD Conference Proceedings 543, Multiple Mechanism Propagation Paths (MMPPs): Their Characteristics and Influence on System Design (Oct), pp. 10-1 to 10-9.

(l) Barrios, A. E. 1994. "A Terrain Parabolic Equation Model for Propagation in the Troposphere," *IEEE Trans. Antennas Propagat.*, vol. 42 (Jan), pp. 90–98.

(m) Naval Oceanographic Office. 1996. "Software Documentation Standards for Environmental System Product Development," Feb.

(n) Barrios, A. E. 1996. "Terrain Parabolic Equation Model (TPEM) Version 1.5 User's Manual." NRaD TD 2898 (Feb), Naval Command, Control and Ocean Surveillance Center RDT&E Division, San Diego, CA.

(o) Sailors, D.B. and A. E. Barrios. 1997. "Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI) Documents." NRaD TD 2963 (May), Naval Command, Control and Ocean Surveillance Center RDT&E Division, San Diego, CA.

3. REQUIREMENTS

3.1 CSCI CAPABILITY REQUIREMENTS

The required APM CSCI propagation model is a range-dependent true hybrid model that uses the complimentary strengths of both Ray Optics (RO) and Parabolic Equation (PE) techniques to calculate propagation loss both in range and altitude.

The atmospheric volume is divided into regions that lend themselves to the application of the various propagation loss calculation methods. Figure 1 illustrates these regions.

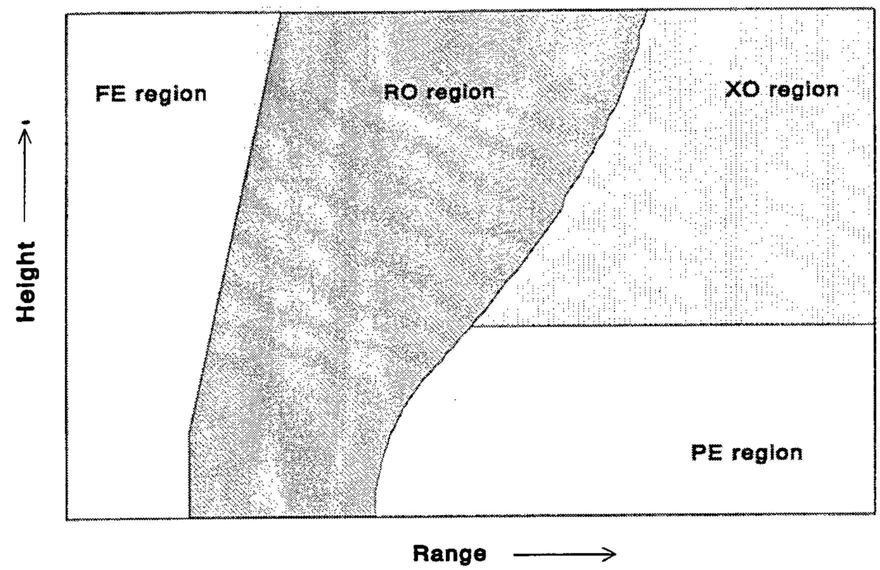

Figure 1. APM calculation regions.

For antenna elevation angles above 5 degrees or for ranges less than approximately 2.5 kilometers (km), a flat earth (FE) RO model is used. In this region, only receiver height is corrected for average refraction and earth curvature.

Within the RO region (as defined by a limiting ray), propagation loss is calculated from the mutual interference between the direct-path and surface-reflected ray components using the refractivity profile at zero range. Full account is given to focusing or de-focusing along both direct and reflected ray paths and to the integrated optical path length difference between the two ray paths to give precise phase difference and, hence, accurate coherent sums for the computation of propagation loss.

For the low-altitude region beyond the RO region, a PE approximation to the Helmholtz full wave equation is employed. The PE model allows for range-dependent refractivity profiles and variable terrain along the propagation path and uses a split-step Fourier method for the solution of the PE. The PE model is run in the minimum region required to contain all terrain and trapping layer heights.

For the area beyond the RO region but above the PE region, an extended optics region (XO) is defined. Within the XO region, RO methods that are initialized by the PE solution from below, are used.

APM will run in three "execution" modes depending on environmental inputs. APM will use the FE, RO, XO, and PE models if the terrain profile is flat for the first 2.5 km and if the antenna height is less than or equal to 100 m. It will use only the XO and PE models if the terrain profile is *not* flat for the first 2.5 km and if the antenna height is less than or equal to 100 m. APM will use only the PE model if the antenna height is greater than 100 m, regardless of terrain profile.

The APM CSCI allows for horizontal and vertical antenna polarization, finite conductivity based on user-specified ground composition and dielectric parameters, and the complete range of

EM system parameters and most antenna patterns required by TESS-NC. APM also allows for gaseous absorption effects in all submodels and computes troposcatter losses within the diffraction region and beyond.

The program flow of the required APM CSCI is illustrated in figure 2. Note that the APM CSCI is shown within the context of a calling CSCI application such as one that generates a coverage or loss diagram. The efficient implementation of the APM CSCI will have far-reaching consequences upon the design of an application CSCI beyond those mentioned in Section 3.10. For example, figure 2 shows checking for the existence of a previously created APM output file prior to the access of the APM CSCI. The application CSCI will have to consider if the atmospheric or terrain environment has changed since the APM output file was created or if any new height or range requirement is accommodated within the existing APM CSCI output file. Because these and many more considerations are beyond the scope of this document to describe, an application CSCI designer should work closely with the APM CSCI development agency in the implementation of the APM CSCI. Figures 2 through 5 illustrate the program flow for the main compute software components (CSC), APMINIT CSC, APMSTEP CSC, XOINIT CSC, XOSTEP CSC, and the APMCLEAN CSC, respectively.

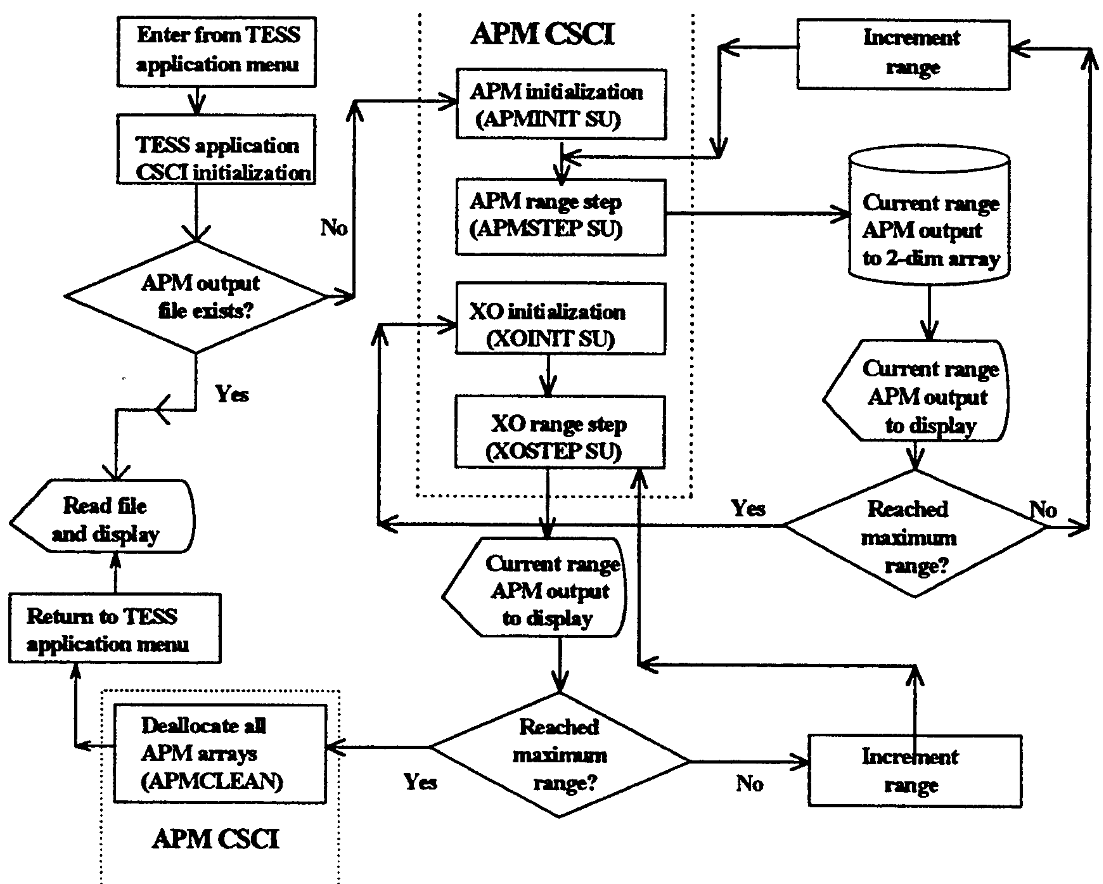

Figure 2. APM CSCI program flow.

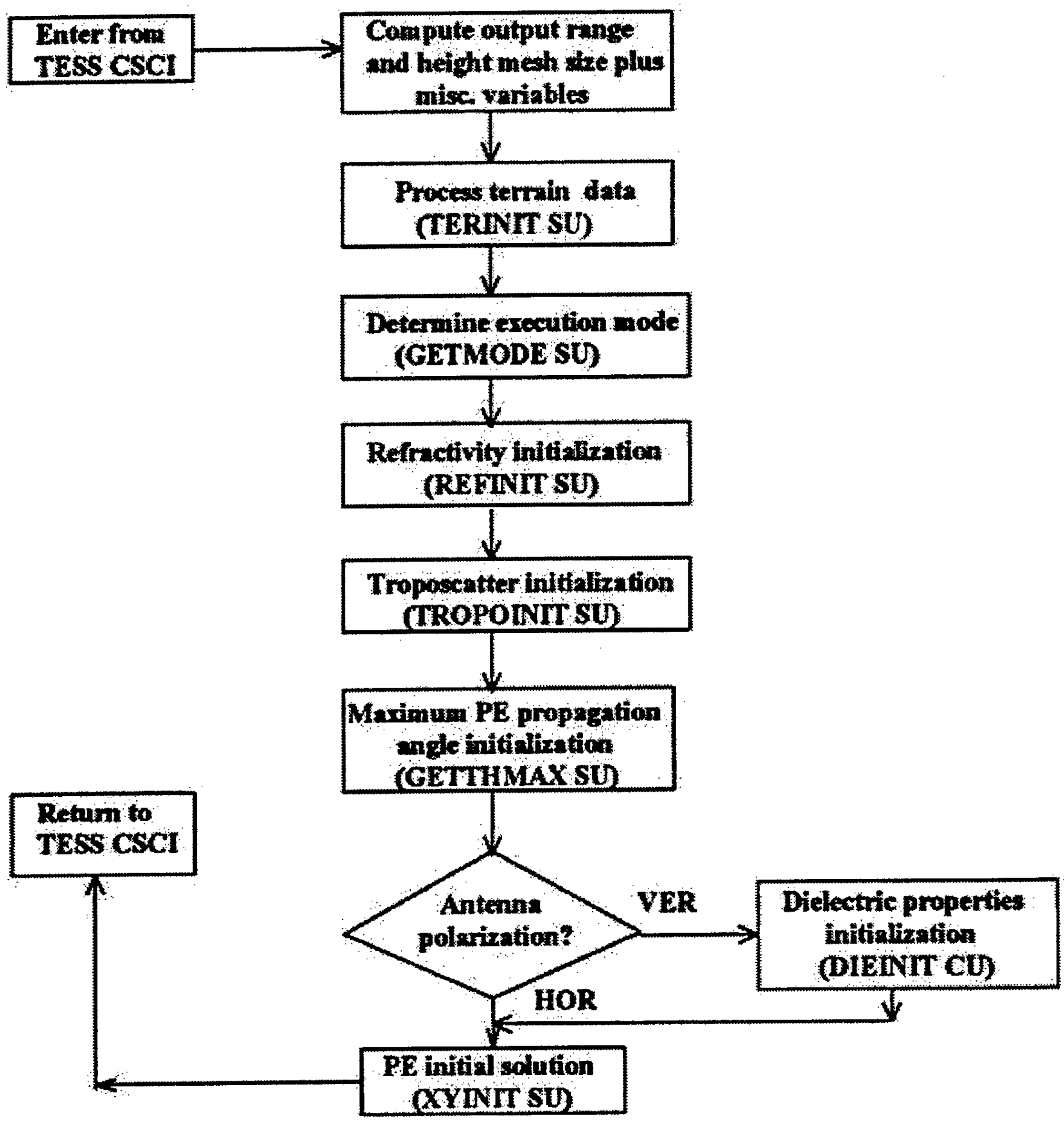

Figure 3. APMSTEP CSC program flow.

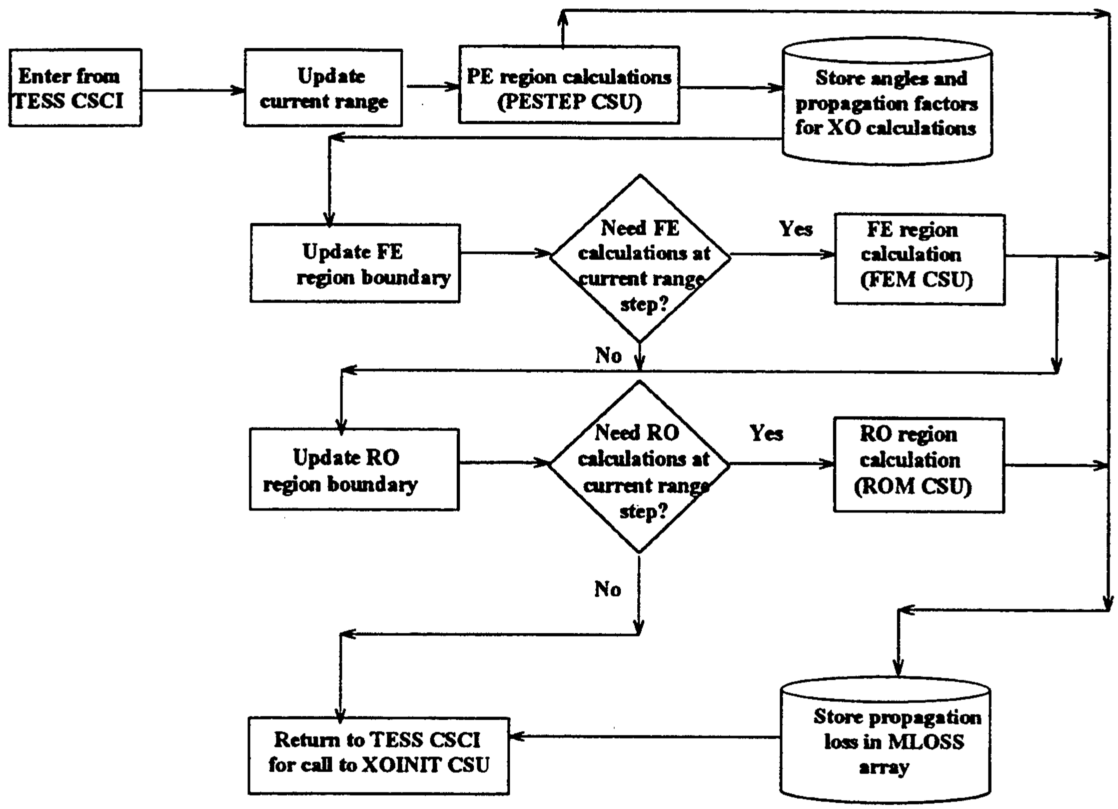

Figure 4. APMSTEP CSC program flow.

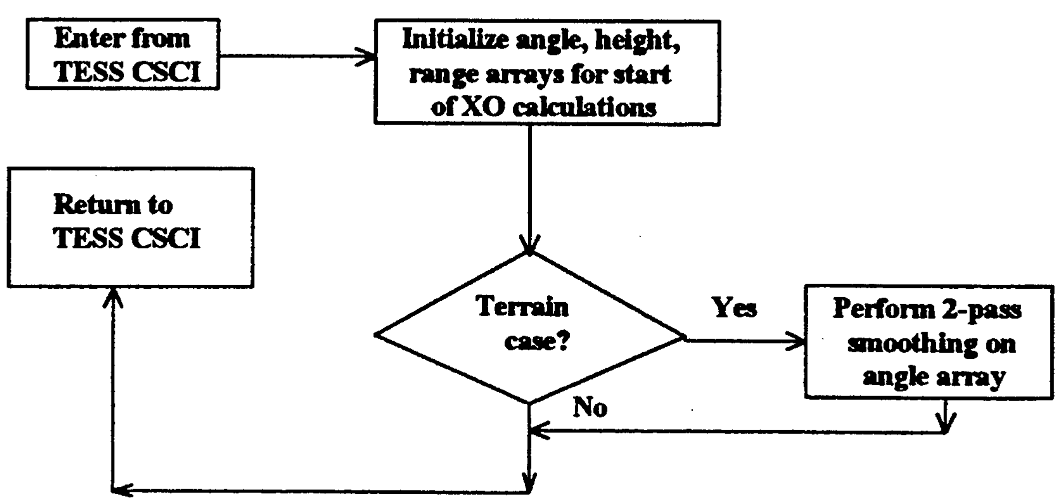

Figure 5. XOINIT CSC program flow.

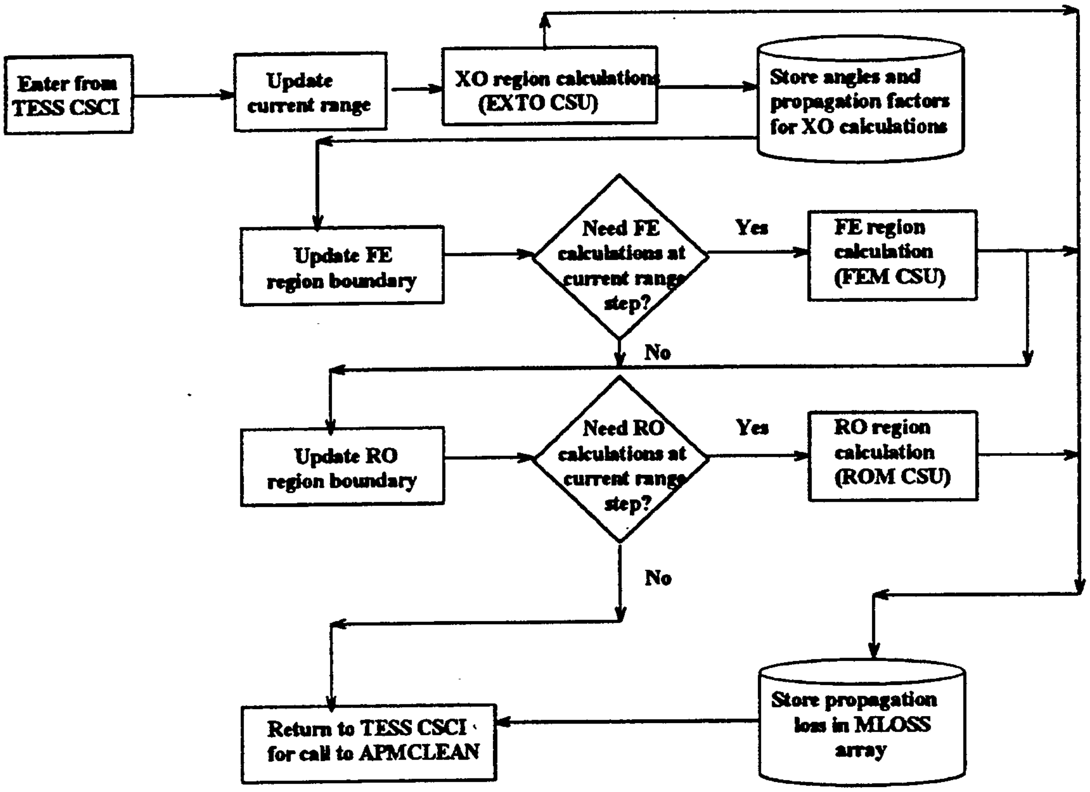

Figure 6. XOSTEP CSC program flow.

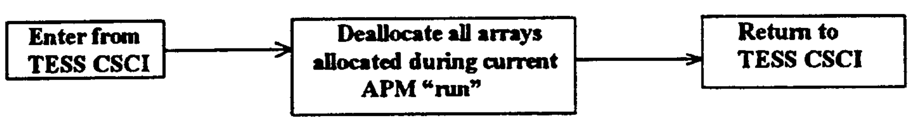

Figure 7. APMCLEAN CSC program flow.

The APM CSCI is divided into five main computer software components (CSCs) and 40 additional software units (SUs). The name, purpose, and a general description of processing required for each SU follows.

3.1.1 Advance Propagation Model Initialization (APMINIT) CSC

The APMINIT CSC interfaces with various SUs for the complete initialization of the APM CSCI.

The atmospheric volume must be "covered" or resolved with a mesh of calculation points that will normally exceed the height/range resolution requirements of the particular application of the APM CSCI. Upon entering the APMINIT CSC, a range and height mesh size per the APM CSCI output point is calculated from the number of APM outputs and the maximum CSCI range and height.

The terrain profile is initially examined and a range increment determined if it is found that range/height points are provided in fixed range increments. The minimum terrain height is determined, and then the entire terrain profile is adjusted by this height so that all internal calculations are referenced to this height. This is done to maximize the PE transform calculation volume.

A GETMODE SU is referenced to determine if the APM CSCI will execute in a full hybrid mode, a partial hybrid mode, or PE-only mode.

A REFINIT SU is referenced to initialize the TESS-NC CSCI specified modified refractivity and also to test for valid environment profiles. A PROFREF SU adjusts the environment profiles by the internal reference height, and a INTPROF SU defines the modified refractivity at all PE vertical mesh points.

To automatically determine the maximum PE calculation angle, a GETTHMAX SU is referenced. This determines, via ray tracing, the minimum angle for which adequate coverage can be given with the specified terrain and environment profile. A FFTPAR SU is referenced to determine the fast Fourier transform (FFT) size for the calculated angle and to initialize data elements within the PE region that are dependent on the size of the FFT. The minimum size for the FFT is determined from the Nyquist criterion.

A PE starting SU (XYINIT) and an antenna pattern factor SU (ANTPAT) are referenced by the XYINIT SU to generate a first solution to the PE. A FFT SU is referenced for data elements required in obtaining the PE's starting solution. If vertical polarization is specified, then additional calculations are performed in the starter solution using Kuttler and Dockery's mixed transform method (reference h). In this case, a DIEINIT SU is used to initialize dielectric ground constants. For general ground types, the permittivity and conductivity are calculated as a function of frequency from curve fits to the permittivity and conductivity graphs shown in recommendations and reports of the International Radio Consulting Committee (reference d).

If running in a full hybrid mode, a FILLHT SU is referenced to determine the heights at each output range separating the FE, RO, and PE calculation regions. If running in a partial hybrid or PE-only mode, then the heights at each output range are determined, below which propagation loss solutions are valid. No propagation loss solutions are provided above these heights for those execution modes.

Finally, a PHASE1 SU is referenced to initialize the free-space propagator array, and a PHASE2 SU is referenced (for a range-independent environment profile) to initialize the environment propagator array.

3.1.1.1 Allocate Arrays APM (ALLARRAY_APM) SU. The ALLARRAY_APM SU allocates and initializes all dynamically dimensioned arrays associated with APM terrain, refractivity, troposcatter, and general variable arrays.

3.1.1.2 Allocate Array PE (ALLARRAY_PE) SU. The ALLARRAY_PE SU allocates and initializes all dynamically dimensioned arrays associated with PE calculations.

3.1.1.3 Allocate Array XO (ALLARRAY_XO) SU. The ALLARRAY_XO SU allocates and initialize all dynamically dimensioned arrays associated with XO calculations.

3.1.1.4 Antenna Pattern (ANTPAT) SU. The ANTPAT SU calculates a normalized antenna gain (antenna pattern factor) for a specified antenna elevation angle.

From the antenna beam width, elevation angle (an angle for which the antenna pattern factor is desired), and the antenna radiation pattern type, an antenna factor is calculated.

3.1.1.5 Dielectric Initialization (DIEINIT) SU. The DIEINIT SU determines the conductivity and relative permittivity as functions of frequency in megahertz based on general ground composition types.

3.1.1.6 Fast-Fourier-Transform (FFT) SU. The FFT SU separates the real and imaginary components of the complex PE field into two real arrays and then references the SINFFT SU that transforms each portion of the PE solution.

3.1.1.7 FFT Parameters (FFTPAR) SU. The purpose of the FFTPAR SU is to determine the required transform size based on the maximum PE propagation angle and the maximum height needed. If running in full or partial hybrid modes, the maximum height needed is the height necessary to encompass at least 20 percent above the maximum terrain peak along the path or the highest trapping layer specified in the environment profiles, whichever is greater. If running in a PE-only mode, the maximum height needed is the specified maximum output height.

For computational efficiency reasons, an artificial upper boundary must be established for the PE solution. To prevent upward propagating energy from being "reflected" downward from this boundary and contaminating the PE solution, the PE solution field strength should be attenuated or "filtered" above a certain height to ensure that the field strength just below this boundary is reduced to zero.

The total number of vertical points for which a transformation will be computed is determined. This term is also referred to as the FFT size. The filtering boundary height is also determined.

3.1.1.8 Fill Height Arrays (FILLHT) SU. The FILLHT SU calculates the effective earth radius for an initial launch angle of 5° and fills an array with height values at each output range of the limiting submodel, depending on which mode is used. If running in a full hybrid mode, then the array contains height values at each output range separating the FE from the RO region. If running in partial hybrid or PE-only modes, then the array contains those height values at each output range at which the initial launch angle has been traced to the ground or surface. These height values represent the separating region where, above that height, valid loss is computed, and below that height, no loss is computed. This is done so that only loss values that fall within a valid calculation region are output.

3.1.1.9 Gaseous Absorption (GASABS) SU. The GASABS SU computes the specific attenuation based on air temperature and absolute humidity. This SU is based on CCIR (International Telecommunication Union, International Radio Consultative Committee, now the ITU-R) Recommendation 676-1, "Attenuation by Atmospheric Gases in the Frequency Range 1-350 GHz."

3.1.1.10 Get Alpha Impedance (GETALN) SU. The GETALN SU computes the impedance term in the Leontovich boundary condition, and the complex index of refraction for finite conductivity and vertical polarization calculations. These formulas follow Kuttler and Dockery's method (reference h).

3.1.1.11 Get Mode (GETMODE) SU. The GETMODE SU determines what "execution" mode APM will run based on environmental inputs for the current application.

3.1.1.12 Get Maximum Angle (GETTHMAX) SU. The GETTHMAX SU performs an iterative ray trace to determine the minimum angle required (based on the reflected ray) in obtaining a PE solution. The determination of this angle will depend on the particular mode of execution. For the full and partial hybrid modes, a ray is traced up to a height that exceeds at least 20 percent above the maximum terrain peak along the path or the highest trapping layer specified in the environment profiles, whichever is greater. For the PE-only mode, a ray is traced for all heights up to the maximum output height. The maximum PE propagation angle is then determined from the local maximum angle of the traced ray.

3.1.1.13 Interpolate Profile (INTPROF) SU

The INTPROF SU performs a linear interpolation vertically with height on the refractivity profile. Interpolation is performed at each PE mesh height point.

3.1.1.14 Free-Space Propagator Phase Term (PHASE1) SU. The PHASE1 SU initializes the free-space propagator array for subsequent use in the PESTEP SU. The propagator term is computed at each PE angle, or p-space, mesh point using the wide-angle propagator. Finally, a filter, or attenuation function (frequently called "window"), is applied to the upper one-quarter of the array corresponding to the highest 1/4 of the maximum propagation angle.

3.1.1.15 Environmental Propagator Phase Term (PHASE2) SU. The PHASE2 SU calculates the environmental phase term for an interpolated environment profile. This environmental phase term is computed at each PE height, or z-space, mesh point. Finally, a filter, or attenuation function (frequently called "window"), is applied to the upper 1/4 of the mesh points corresponding to the highest 1/4 of the calculation height domain.

3.1.1.16 Profile Reference (PROFREF) SU. The PROFREF SU adjusts the current refractivity profile so that it is relative to a reference height. The reference height is initially the minimum height of the terrain profile. Upon subsequent calls from the PESTEP SU, the refractivity profile is adjusted by the local ground height at each PE range step.

3.1.1.17 Refractivity Initialization (REFINIT) SU. The REFINIT SU checks for valid environmental profile inputs and initializes all refractivity arrays.

The environmental data are checked for a range-dependent profile and tested to determine if the range of the last profile entered is less than the maximum output range specified. If so, an integer error flag is returned and the SU exited, depending on the values of logical error flags set in the TESS-NC CSCI itself.

The REFINIT SU also tests for valid refractivity level entries for each profile. If the last gradient in any profile is negative, it returns an integer error flag and the SU is exited. If no errors are detected, the REFINIT SU then extrapolates the environmental profiles vertically to 1000 km in height. Extrapolation is not performed horizontally from the last provided profile; rather, the last provided environment profile is duplicated at 10^7 km in range. This duplication of profiles is done by the REFINTER SU.

3.1.1.18 Sine Fast-Fourier Transform (SINFFT) SU. A function with a common period, such as a solution to the wave equation, may be represented by a series consisting of sines and cosines. This representation is known as a Fourier series. An analytical transformation of this function, known as a Fourier transform, may be used to obtain a solution for the function.

The solution to the PE approximation to Maxwell's wave equation is to be obtained by using such a Fourier transformation function. The APM CSCI requires only the real-valued sine transformation in which the real and imaginary parts of the PE equation are transformed separately. A Fourier transformation for possible use with the APM CSCI is described by Bergland (reference a) and Cooley, Lewis, and Welsh (reference b).

3.1.1.19 Terrain Initialization (TERINIT) SU. The TERINIT SU examines and initializes terrain arrays for subsequent use in PE calculations. It tests for and determines a range increment if it is found that range/height points are provided in fixed range increments. The minimum terrain height is determined and the entire terrain profile is adjusted so that all internal calculations are referenced to this height. This is done to maximize the PE transform calculation volume.

3.1.1.20 Troposcatter Initialization (TROPOINIT) SU. The TROPOINIT SU initializes all variables and arrays needed for subsequent troposcatter calculations. The tangent range and tangent angle are determined from the source and from all receiver heights and stored in arrays.

3.1.1.21 Starter Field Initialization (XYINIT) SU. The XYINIT SU calculates the complex PE solution at range zero.

Several constant terms that will be employed over the entire PE mesh are calculated. These are the angle difference between mesh points in p-space and a height-gain value at the source (transmitter).

For each point in the PE p-space mesh, the following steps are performed:

1. The antenna pattern ANTPAT SU is referenced to obtain an antenna pattern factor for both a direct-path ray and a surface-reflected ray. Since the PE starting solution makes a flat-earth assumption, the direct-path ray elevation angle is used in place of the surface-grazing angle.

2. The complex portions of the PE solution are determined from the antenna pattern factors, elevation angle, and gain. The initial field assumes the source is horizontally polarized over a perfectly conducting ground.

3.1.2 Advanced Propagation Model Step (APMSTEP) CSC. The APMSTEP CSC advances the entire APM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range. At this current range, APM calculations will be made within the vertical (up to the maximum PE height region) by accessing the appropriate region's SUs.

The current output range is determined. The PESTEP SU is referenced to obtain the PE portion of the propagation loss at this new range.

If running in full hybrid mode, then based upon a height array index used within the FE region, a determination is made for the necessity to include FE propagation calculations. If so, the FEM SU is referenced to obtain the FE portion of the propagation loss. If a FE calculation is made, the maximum height index for the RO region is adjusted (with the minimum height index corresponding to the maximum height index of the PE region), and the ROM SU is referenced to obtain the RO portion of the propagation loss at the current range. FE and RO propagation loss will be computed only up to the range at which XO calculations will be performed.

If running in partial hybrid or PE-only modes, then only the PESTEP SU will be referenced to obtain the PE portion of the propagation loss at this new range. For the partial hybrid mode, the maximum height will correspond to the maximum height of the PE calculation region. For the PE-only mode, the maximum height corresponds to the maximum specified coverage height.

Finally, absorption loss is computed for the current range and added to the propagation loss at all heights.

3.1.2.1 Calculate Propagation Loss (CALCLOS) SU. The CALCLOS SU determines the propagation loss from the complex PE field at each output height point at the current output range.

The local ground height at the current output range is determined. All propagation loss values at output height points up to the local ground height are then set to zero. The first valid loss point is determined corresponding to the first output height point above the ground height. Next, the last valid loss point is determined based on the smaller of the maximum output height or the height traced along the maximum PE propagation angle to the current output range.

From the height of the first valid loss point to the height of the last valid loss point, the GETPFAC SU is referenced to obtain the propagation factor in dB (field strength relative to free space) at all corresponding output heights at the previous and current PE ranges. Then, for each valid output height, horizontal interpolation in range is performed to obtain the propagation factor at the current output range. From the propagation factor and the free-space loss, the propagation loss at each valid output height is then determined, with the propagation loss set to -1 for all output height points above the last valid output height but less than the maximum output height.

If running in full or partial hybrid modes, the propagation factor at the top of the PE region is determined at every output range and stored in an array for future reference in XO calculations. If troposcatter calculations are desired, the TROPO SU is referenced with the results added to the propagation loss array. All loss values returned to the TESS-NC CSCI at this point are in centibels (10 cB = 1 dB).

3.1.2.2 DOSHIFT SU. The DOSHIFT SU shifts the complex PE field by the number of bins, or PE mesh heights corresponding to local ground height.

The number of bins to be shifted are determined. The PE solution is then shifted downward if the local ground is currently at a positive slope, and upward if the local ground is at a negative slope.

3.1.2.3 Flat Earth Model (FEM) SU. The FEM SU computes propagation loss at a specified range based upon flat-earth approximations. Receiver heights are corrected for earth curvature and average refraction based on twice the effective earth radius computed in the FILLHT SU. The following steps are performed for each APM output height.

1. The path lengths and elevation angles for both the direct-path and surface-reflected path, along with the grazing angle, are computed from simple right triangle calculations. Using the two elevation angles, the ANTPAT SU is referenced to obtain an antenna pattern factor for each angle. Using the grazing angle, the GETREFCOEF SU is referenced to obtain the magnitude and phase lag of the surface reflection coefficient.
2. From the path length difference, the phase lag of the surface reflected ray, and the wave number, a total phase lag is determined. Using the total phase lag, the magnitude of the surface reflection coefficient and the two antenna pattern factors, the two ray components are coherently summed to obtain a propagation factor. The propagation factor, together with the free-space propagation loss and path length difference of the direct-path ray are used to compute the propagation loss.

3.1.2.4 Free-Space Range Step (FRSTP) SU. The FRSTP SU propagates the complex PE solution field in free space by one range step.

The PE field is transformed to p-space and then multiplied by the free space propagator. Before exiting, the PE field is transformed back to z-space. Both transforms are performed using a FFT SU.

3.1.2.5 FZLIM SU. The FZLIM SU determines both the propagation factor (in dB) and the outgoing propagation angle at the top of the PE calculation region. These values, along with the corresponding PE range, are stored for future reference by the XOINIT SU.

The GETPFAC SU is referenced to determine the propagation factor at the last height mesh point in the valid part of the PE region. The propagation factor, along with the range and the local ray angle (determined from the ray traced separating the RO and PE regions), is stored if this is the first call to the FZLIM SU. The SPECEST SU is then referenced to determine the outgoing propagation angle. Depending on the change of angles from one range step to the next, the calculated outgoing angle will be limited. The storage array counter is incremented and the outgoing angle stored.

Before exiting, the SAVEPRO SU is referenced to store the refractivity profiles from the top of the PE region to the maximum specified coverage height.

3.1.2.6 Get Propagation Factor (GETPFAC) SU. The GETPFAC SU determines the propagation factor at the specified height in dB.

A vertical interpolation with height on the PE solution field is performed to obtain the magnitude of the field at the desired output height point. An additional calculation is made and the propagation factor is then returned in dB.

3.1.2.7 Get Reflection Coefficient (GETREFCOEF) SU. The GETREFCOEF SU calculates the complex surface reflection coefficient, along with the magnitude and phase angle.

The complex reflection coefficient is computed from a specified grazing angle and is based on the Fresnel reflection coefficient equations for vertical and horizontal polarization. The magnitude and phase angle are determined from the complex reflection coefficient. If the polarization is horizontal and the frequency is greater than 300 MHz, the magnitude of the reflection coefficient is set to 1 and the phase angle is set to π.

3.1.2.8 Parabolic Equation Step (PESTEP) SU. The PESTEP SU advances the PE solution one output range step, referencing various SUs to calculate the propagation loss at the current output range.

The next output range is determined and an iterative loop begun to advance the PE solution such that for the current PE range, a PE solution is calculated from the solution at the previous PE range. This procedure is repeated until the output range is reached.

At each PE range step, the local ground height is determined and the PE field is "shifted" by the number of bins, or PE mesh height points, corresponding to the local ground height. This is performed in the DOSHIFT SU.

If using vertical polarization and the current ground type has changed from the previous one, a GETALN SU is referenced to determine the impedance term and all associated variables used for the mixed transform calculations.

If the APM CSCI is being used in a range-dependent mode, that is, more than one profile has been input; or a terrain profile is specified, the REFINTER SU is referenced to compute a new modified refractive index profile adjusted by the local ground height at the current range. The PHASE2 SU is then referenced to compute a new environmental phase term using this new refractivity profile.

Using a FRSTP SU, the PE solution is transformed to p-space, advanced by the free space propagator array, and transformed back to z-space. The environmental phase term is then applied to obtain the new final PE solution at the current range. Once all calculations are made to determine the PE field at the current PE range, the FZLIM SU is referenced to determine and store the outgoing propagation factor and propagation angle at the top of the PE region. The FZLIM SU is only referenced if running in full or partial hybrid modes. Finally, a CALCLOS SU is referenced to obtain the propagation loss at the desired output heights at the current output range.

3.1.2.9 Ray Trace (RAYTRACE) SU. Using standard ray trace techniques, a ray is traced from a starting height and range with a specified starting elevation angle to a termination range. As the ray is being traced, an optical path length difference and a derivative of range with respect to elevation angle are being continuously computed. If the ray should reflect from the surface, a grazing angle is determined. Upon reaching the termination range, a terminal elevation angle is determined along with a termination height.

3.1.2.10 Refractivity Interpolation (REFINTER) SU. The REFINTER SU interpolates both horizontally and vertically on the modified refractivity profiles. Profiles are then adjusted so they are relative to the local ground height.

If range-dependent refractive profiles have been specified, horizontal interpolation to the current PE range is performed between the two neighboring profiles. A REMDUP SU is referenced to remove duplicate refractivity levels, and the PROFREF SU is then referenced to adjust the new profile relative to the internal reference height corresponding to the minimum height of the terrain profile. The PROFREF SU is referenced once more to adjust the profile relative to the local ground height, and upon exit from the PROFREF SU, the INTPROF SU is referenced to interpolate vertically on the refractivity profile at each PE mesh height point.

3.1.2.11 Remove Duplicate Refractivity Levels (REMDUP) SU. The REMDUP SU removes any duplicate refractivity levels in the currently interpolated profile.

3.1.2.12 Ray Optics Calculation (ROCALC SU). The ROCALC SU computes the RO components that will be needed in the calculation of propagation loss at a specified range and height within the RO region. These components are the amplitudes for a direct-path and surface-reflected ray, and the total phase lag angle between the direct-path and surface-reflected rays.

A test is made to determine if this is the first RO calculation. If an initial calculation is needed, the height, range, and elevation angle array indices are set to initial conditions. If not, the array indices are incremented from the previous RO calculation.

The following steps are performed for each series of vertical grid points, in a manner that ensures that RO calculations have been performed at ranges that span the current range of interest. The vertical grid points are taken in order beginning with the one with greatest height.

1. Using a Newton iteration method with a varying elevation angle, the RAYTRACE SU is referenced to find a direct-path ray and a surface-reflected ray which will originate at the transmitter height and terminate at the same grid point. Should a direct or reflected ray not be found to satisfy the condition, or should the computed grazing angle exceed the grazing angle limit, the height array index is adjusted to redefine the lower boundary of the RO region. Should the ray trace conditions be satisfied, the RAYTRACE SU will provide a terminal elevation angle, a derivative of range with respect to elevation angle, a path length, and for the surface-reflected ray, a grazing angle.

2. Using the final direct-path ray and surface-reflected ray elevation angles obtained from the Newton iteration method, the ANTPAT SU is referenced to obtain an antenna pattern factor for each angle. The GETREFCOEF SU is referenced to obtain the amplitude and phase lag angle of the surface reflection coefficient.

3. Using the antenna pattern factors, path length differences, and surface-reflection coefficients, the necessary RO components defined in the first paragraph above are calculated.

3.1.2.13 Ray Optics Loss (ROLOSS) SU. The ROLOSS SU calculates the propagation loss values at a specified range and height based upon the components of magnitude for a direct-path and surface-reflected ray and the total phase lag angle between the two rays as determined by the ROCALC SU.

For purposes of computational efficiency, an interpolation from the magnitude and total phase lag arrays, established by the ROCALC SU, is made to obtain these three quantities at each APM vertical output mesh point within the RO region.

From the interpolated phase lag and ray amplitudes, a propagation factor is calculated which is used, in turn, with the free-space propagation loss to obtain a propagation loss at each vertical APM output point.

3.1.2.14 Ray Optics Model (ROM) SU. The ROM SU provides a one-call routine for RO calculations.

The SU references the ROCALC SU and determines the loss at specified height output points by referencing the ROLOSS SU.

3.1.2.15 Save Profile (SAVEPRO) SU. The SAVEPRO SU stores refractivity profiles at each PE range step from the top of the PE region to the maximum user-specified height. This is only done if running in full or partial hybrid modes.

The refractivity height level which just exceeds the PE region height limit, is determined. From this level upward, all heights, M-units, and gradients are stored.

3.1.2.16 Spectral Estimation (SPECEST) SU. The SPECEST SU determines, via spectral estimation, the outward propagation angle at the top of the PE calculation region.

The upper 8 (if running smooth surface case) or 16 (if running terrain case) bins of the complex PE field at the current PE range are separated into their real and imaginary components. The upper ¼ of this portion of the field is then filtered and zero-padded to 256 points. It is then transformed to its spectral components via a reference to the SINFFT SU. The amplitudes of the spectral field are then determined and a three-point average is performed. The peak of the 256-point field is then found and the outgoing propagation angle is determined from the peak value.

3.1.2.17 Troposcatter (TROPO) SU. The TROPO SU determines the loss due to troposcatter and to compute the appropriate loss from troposcatter and propagation loss.

The current output range is updated and the tangent angle from the source to the current output range is initialized. For all output receiver heights at the current output range, the following procedure is performed.

1. If the current output range is less than the minimum diffraction field range for a particular receiver height, then the SU is exited and no troposcatter loss is computed.
2. The tangent angle from the receiver height is determined.
3. The common volume scattering angle is determined and calculations are performed to obtain the loss due to troposcatter.
4. Troposcatter loss is compared to propagation loss. If the difference between the propagation loss and troposcatter loss is less than 18 dB, then the corresponding power levels of the two loss values are added. If the difference is greater than 18 dB, then the lesser of the two losses is used.

3.1.3 Extended Optics Initialization (XOINIT) CSC

The purpose of the XOINIT SU is to initialize the range, height, and angle arrays in preparation for the XOSTEP CSC.

Upon entering, all dynamically allocated arrays used for XO calculations are allocated and initialized to 0. The ranges and angles previously stored from referencing the FZLIM SU are now used to initialize the range and angle arrays. A 10-point smoothing average on the angle array is performed twice via reference to the SMOOTH SU. Upon exiting, the height array and initial height index for start of XO calculations are initialized.

3.1.3.1 Smooth (SMOOTH) SU. The SMOOTH SU performs an n-point average smoothing on any array passed to it.

3.1.4 Extended Optics Step (XOSTEP) CSC

The XOSTEP CSC advances the APM CSCI algorithm one output range step from the top of the PE calculation region to the maximum output height specified, referencing various SUs to calculate the propagation loss at the current output range.

Upon entering the XOSTEP CSC, the current output range is determined. The EXTO SU is referenced to obtain the XO portion of the propagation loss at this new range.

If running in full hybrid mode, based upon a height array index used within the FE region, it is determined if it is necessary to include FE propagation calculations. If necessary, the FEM SU is referenced to obtain the FE portion of the propagation loss. If a FE calculation is made, the maximum height index for the RO region is adjusted (with the minimum height index corresponding to the maximum height index of the PE region), and the ROM SU is referenced to obtain the RO portion of the propagation loss at the current range.

If running in partial hybrid mode, then only the EXTO SU is referenced to obtain the XO portion of the propagation loss at this new range. The maximum height will correspond to the maximum user-specified coverage height.

Finally, absorption loss is computed for the current range and added to the propagation loss at all heights.

3.1.4.1 Extended Optics (EXTO) SU. The EXTO SU calculates propagation loss, based on extended optics techniques, at the current output range.

Upon entering, array indices for the current range, height, and angle arrays are initialized. A ray trace is then performed for all rays from the last output range to the current output range. The current heights are then sorted, along with their corresponding propagation factors. The propagation loss is then determined at each output receiver height by interpolation on the terminal heights of the traced rays.

Upon exiting, a reference to the TROPO SU provides any troposcatter losses and this is added to the loss array.

3.1.5 Advanced Propagation Model Clean (APMCLEAN) CSC

The APMCLEAN CSC deallocates all dynamically dimensioned arrays used in one complete run of APM calculations.

3.2 CSCI EXTERNAL INTERFACE REQUIREMENTS

The APM CSCI is accessed, through the APMINIT CSC, by a subroutine call from the TESS-NC CSCI which should provide, as global data elements, the values specified in table 1 through 4.

The APM CSCI external data elements (i.e., data which must be provided by the calling TESS-NC CSCI prior to the APM CSCI execution may be divided into four classifications). The first classification is external data related to the atmospheric environment, specified within table 1; the second is data related to the EM system, specified in table 2; the third is data related to the implementation of the APM CSCI by the TESS-NC CSCI, specified in table 3; and the fourth is data related to the terrain information, specified in table 4. Each table lists the type, units, and bounds of each data element. Table 5 specifies the output data of the APM CSCI model.

Table 1. APM CSCI environmental data element requirements.

Name	Description	Type	Units	Bounds
refmsl	Profile modified refractivity (dynamically allocated) array referenced to mean sea level	real	M	≥ 0.0[a]
hmsl	Profile height (dynamically allocated) array	real	meters	See note b
n_{prof}	Number of profiles	integer	N/A	≥ 1
lvlp	Number of profile levels	integer	N/A	≥ 2
rngprof	Dynamically allocated array of ranges to each profile	real	meters	≥ 0.0
abs_{hum}	Surface absolute humidity	real	g/m^3	0 to 50[c]
t_{air}	Surface air temperature	real	°C	-20 to 40[c]
γ_a	Surface specific attenuation	real	dB/km	≥0.0
i_{extra}	Extrapolation flag for refractivity profiles entered below mean sea level	integer	N/A	0 or 1

[a]Couplets of height and modified refractivity associated with that height are referred to within this document as an environmental profile.

[b]All heights in the refractivity profile must be steadily increasing.

[c]The CCIR gaseous absorption model implemented within APM provides a ±15% accuracy for absolute humidity and surface air temperature within these bounds. While values beyond these limits are allowed within APM, it should be noted this may result in less accurate attenuation rates calculated.

Table 2. APM CSCI external EM System data element requirements

Name	Description	Type	Units	Bounds
μ_{bw}	Antenna vertical beam width	real	degree	.5 to 45
μ_o	Antenna elevation angle	real	degree	-50.0 to 50.0
f_{MHz}	EM system frequency	real	MHz	100.0 to 20,000.0
i_{pat}	Antenna pattern 1 = Omni-directional 2 = Gaussian 3 = Sine (X)/X 4 = Cosecant-squared 5 = Generic height-finder 6 = User-defined height-finder	integer	N/A	1 to 6
i_{pol}	Antenna polarization 0 = Horizontal 1 = Vertical	integer	N/A	0 or 1
ant_{ht}	Antenna height above local ground at range 0.0 m	real	meters	≥ 1.0
hfang	Dynamically allocated user-defined height-finder power reduction angle array	real	degree	0.0 to 90.0
hffac	Dynamically allocated user-defined power reduction factor array	real	N/A	0.0 to 1.0
n_{facs}	Number of power reduction angles/factors for user-defined height finder radar	integer	N/A	1 to 10

Table 3. APM CSCI external implementation constants.

Name	Description	Type	Units	Bounds
n_{rout}	Number of range output points for a particular application of APM	integer	N/A	≥ 1
n_{zout}	Number of height output points for a particular application of APM	integer	N/A	≥ 1
lerr6	Logical flag to allow for error -6 to be bypassed	logical	N/A	'.true.' or '.false.' [a]
lerr12	Logical flag to allow for error -12 to be bypassed	logical	N/A	'.true.' or '.false.' [a]
i_{tropo}	Flag to include troposcatter calculations (0 = no, 1 = yes)	integer	N/A	0 or 1
r_{max}	Maximum range output for a particular application of APM	real	meters	$\geq$ 5000.0 [b]
h_{min}	Minimum height output for a particular application of APM	real	meters	$\geq$ 0.0 [c]
h_{max}	Maximum height output for a particular application of APM	real	meters	$\geq$ 100.0 [b]

[a] refer to section 3.5.1 for a complete description.

[b] refer to section 3.5.2 for a complete description.

[c] refer to section 3.5.3 for a complete description.

Table 4. APM CSCI external terrain data element requirements.

Name	Description	Type	Units	Bounds
terx	Dynamically allocated terrain profile range array	real	meters	≥ 0.0 [a]
tery	Dynamically allocated terrain profile height array	real	meters	≥ 0.0 [a]
i_{tp}	Number of terrain profile points for a particular application of APM	integer	N/A	≥ 2
i_{gr}	Number of ground types for a particular application of APM	integer	N/A	≥ 0.0 [a]
igrnd	Array of ground composition types for a particular application of APM 0 = Sea water 1 = Fresh water 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1° C 6 = Ice at -10° C 7 = User-defined	integer	N/A	0 ≤ *igrnd* ≤ 7 [a]
rgrnd	Dynamically allocated array of ranges for which ground types are applied for a particular application of APM	real	meters	≥ 0.0 [a]
dielec	Dynamically allocated 2-dimensional array of relative permittivity and conductivity for a particular application of APM	real	N/A	>0 [a]

[a] refer to Section 3.5.3 for a complete description.

Table 5. APM CSCI output data element requirements.

Name	Description	Type	Units	Source
i_{xostp}	Index of output range step at which XO model is to be applied	integer	N/A	APMINIT CSC
i_{error}	Integer value that is returned if an error occurs in called routine	integer	N/A	APMINIT CSC XOINIT CSC APMCLEAN CSC
$mloss$	Propagation loss	integer	cB	APMSTEP CSC XOSTEP CSC
j_{start}	Output height index at which valid PE propagation loss values begin	integer	N/A	APMSTEP CSC
j_{end}	Output height index at which valid PE propagation loss values end	integer	N/A	APMSTEP CSC
r_{out}	Current range	real	meters	APMSTEP CSC XOSTEP CSC
j_{xstart}	Output height index at which valid XO propagation loss values begin	integer	N/A	XOINIT CSC
j_{xend}	Output height index at which valid XO propagation loss values end	integer	N/A	XOSTEP CSC

[a]Refer to Section 3.5.1 for a complete description.

3.3 CSCI INTERNAL INTERFACE REQUIREMENTS

Section 3.1 shows the relationship between the APM CSCI and its five CSCs APMINIT, APMSTEP, XOINIT, XOSTEP, and APMCLEAN. The required internal interface between these five CSCs and the APM CSCI is left to the designer. However, table 6 should be used as a guide to the required internal interfaces in the CSCI.

3.4 CSCI INTERNAL DATA REQUIREMENTS

The APM CSCI takes full advantage of Fortran 90 features, utilizing allocatable arrays for all internal and input arrays. This requires the TESS-NC CSCI designer to correctly allocate and initialize all arrays necessary for input to the APM CSCI. The APMCLEAN CSC is provided as part of the APM CSCI and should be called by the TESS-NC application to deallocate all arrays used by the APM CSCI in one complete run.

Due to the computational intensity of the APM CSCI, it may not be necessary or desirable to use the extreme capability of the APM CSCI for all applications. The variables n_{rout} and n_{zout} refer to the desired number of range and height output points for any one particular application, and will be specified when the APMINIT CSC is called.

One of the parameters returned to the TESS-NC application from the APMINIT CSC is i_{error}. This allows greater flexibility in how input data are handled within the TESS-NC application. Table Table 6 lists all possible errors that can be returned.

Table 6. APMINIT SU returned error definitions.

error	Definition
-6	Last range in terrain profile is less than r_{max}. Will only return this error if *lerr6* set to '.true.'
-7	Specified cut-back angles (for user-defined height finder antenna pattern) are not increasing
-8	h_{max} is less than maximum height of terrain profile
-9	Antenna height with respect to mean sea level is greater than maximum height, h_{max}
-10	Beamwidth is less than or equal to zero for directional antenna pattern.
-12	Range of last environment profile given (for range-dependent case) is less than r_{max}. Will only return this error if *lerr12* set to '.true.'
-13	Height of first level in any user-specified refractivity profile is greater than 0. First height must be at mean sea level (0.0) or < 0.0 if below mean sea level
-14	Last gradient in any environment profile is negative
-17	Range points of terrain profile are not increasing
-18	First range value in terrain profile is not 0.
-42	Minimum height input by user, h_{min}, is greater than maximum height, h_{max}

The logical variables, *lerr6* and *lerr12*, when set to '.false.', allow the TESS-NC application to bypass their associated errors as these are not critical to the operation of the APM CSCI.

The APM CSCI provides propagation loss for all heights and ranges when running in a full hybrid mode. When running in a partial hybrid mode, it does provide propagation loss for all heights, but not necessarily for all angles. Finally, it will be limited in both height and angle coverage when running in a PE-only mode. Refer to Section 0 for environmental conditions under which each execution mode is automatically selected.

Absorption by atmospheric gases (oxygen and water vapor) may be important to some applications of the APM CSCI and is controlled by specifying a non-zero value for the absolute humidity, abs_{hum}, and the surface air temperature, t_{air}, or likewise, specifying a non-zero value for the gaseous absorption attenuation rate, γ_a.

A particular application of the APM CSCI may or may not require the consideration of troposcatter effects within the propagation loss calculations. For example, a radar evaluation most likely would not be influenced by troposcatter; while an ESM evaluation would. APM has the feature of including or not including the troposcatter calculation by setting a parameter called i_{tropo}. Setting this parameter to 0 would omit the calculation. Setting this parameter to 1 would include the calculation. For the APM CSCI implementation within the TESS-NC coverage and loss diagram applications, i_{tropo} must be set equal to 1 so as to include the calculation.

3.5 ADAPTATION REQUIREMENTS

3.5.1 Environmental Radio Refractivity Field Data Elements

The radio-refractivity field (i.e., the profiles of M-units versus height) must consist of vertical piece-wise linear profiles specified by couplets of height in meters with respect to mean sea level and modified refractivity (M-units) at multiple arbitrary ranges. All vertical profiles must contain the same number of vertical data points, and be specified such that each numbered data point corresponds to like-numbered points (i.e., features) in the other profiles. The first numbered data point of each profile must correspond to a height of zero mean sea level and the last numbered data point must correspond to a height such that the modified refractivity for all greater heights is well represented by extrapolation using the two highest profile points specified.

With the inclusion of terrain and allowing the terrain profile to fall below mean sea level, refractivity profiles can also be provided in which the first level is less than 0 (or below mean sea level). For a terrain profile that falls below mean sea level at some point, the assumption is that the minimum height may be less than the first height in any refractivity profile specified. Therefore, an extrapolation flag, i_{extra}, must be specified to indicate how the APM CSCI should extrapolate from the first refractivity level to the minimum height along the terrain profile. Setting i_{extra} to 0 will cause the APM CSCI to extrapolate to the minimum height using a standard atmosphere gradient; setting i_{extra} to 1 will cause the APM CSCI to extrapolate to the minimum height using the gradient determined from the first two levels of the refractivity profile.

Within each profile, each numbered data point must correspond to a height greater than or equal to the height of the previous data point. Note that this requirement allows for a profile that contains redundant data points. Note also that all significant features of the refractivity profiles must be specified, even if they are above the maximum output height specified for a particular application of APM.

The TESS-NC CSCI application designer and the TESS-NC operator share responsibility for determining appropriate environmental inputs. For example, a loss diagram may be used to consider a surface-to-surface radar detection problem. Since the operator is interested in surface-to-surface, he may truncate the profile assuming that effects from elevated ducting conditions are negligible. It may be, however, that the elevated duct does indeed produce a significant effect. The operator should ensure, therefore, that the maximum height of the profile allows for the inclusion of all significant refractive features.

This specification allows a complicated refractivity field to be described with a minimum of data points. For example, a field in which a single trapping layer linearly descends with increasing range can be described with just two profiles containing only four data points each, frame (a) of figure 8. In the same manner, other evolutions of refractive layers may be described. Frames (b) and (c) of figure 8 show two possible scenarios for the development of a trapping layer. The scenario of choice is the one which is consistent with the true thermodynamical and hydrological layering of the atmosphere.

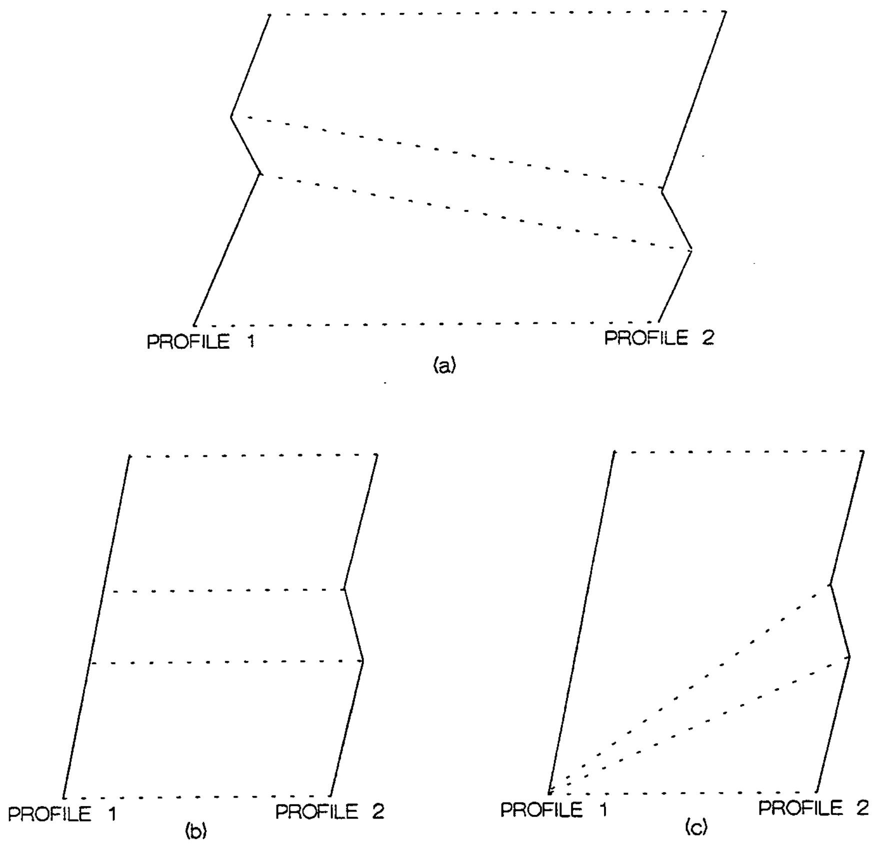

Figure 8. Idealized M-unit profiles (solid) and lines of interpolation (dashed).

Two external implementation data variables applicable to both the TESS-NC operator and to the calling application designer are r_{max}, the maximum APM CSCI output range, and h_{max}, the maximum APM CSCI output height. These two parameters are required by the APM CSCI to determine the horizontal and vertical resolution, respectively, for internal range and height calculations based on the current values of n_{rout} and n_{zout}. Any value of r_{max} and h_{max} is allowed for the convenience of the TESS-NC operator and the calling application designer, provided $r_{max} \geq 5$ km, and $h_{max} \geq 100$ m. For example, the TESS-NC operator may desire a coverage diagram that extends to a range of 500 kilometers (km). In addition to accommodating the desires of the operator, specification of such a convenient maximum range eases the burden for the application designer in determining incremental tick marks for the horizontal axis of the display.

Provided the value of the parameter, *lerr12*, is set to '.false.', if the furthest environment profile range is less than r_{max}, the APM CSCI will automatically create an environment profile at r_{max} equal to the last profile specified, making the environment homogeneous from the range of the last profile specified to r_{max}. For example, a profile is input with an accompanying range of 450 km. If the TESS-

NC operator chooses an r_{max} of 500 km, the APM CSCI will continue loss calculations to 500 km, keeping the refractivity environment homogeneous from 450 to 500 km.

If *lerr12* is set to '.true.' and the furthest environment profile range is less than r_{max}, then an error will be returned in i_{error} from the APMINIT CSC. This is to allow the TESS-NC CSCI application designer greater flexibility in how environment data is handled.

3.5.2 Terrain Profile Data Element

The terrain profile must consist of linear piece-wise segments specified in terms of range/height pairs. All range values must be increasing, and the first terrain height value must be at range zero. General ground composition types can be specified (table 4) along with corresponding ranges over which the ground type is to be applied. If ground type "User Defined" is specified ($igrnd_i = 7$), then numeric values of relative permittivity and conductivity must be given. If horizontal antenna polarization is specified, the APM CSCI will assume perfect conductivity for the entire terrain profile and will ignore any information regarding ground composition. If vertical antenna polarization is specified, then information regarding ground composition must also be specified.

The maximum height, h_{max}, must always be greater than the minimum height, h_{min}. Also, a value of h_{max} must be given such that it is larger than the maximum elevation height along a specified terrain profile.

Provided *lerr6* is set to '.false.', if the furthest range point in the terrain profile is less than r_{max}, the APM CSCI will automatically create a height/range pair as part of the terrain profile at r_{max} with elevation height equal to the last height specified in the profile, making the terrain profile flat from the range of the last profile point specified to r_{max}. For example, a terrain profile is input where the last height/range pair is 50 meters (m) in height with an accompanying range of 95 km. If the TESS-NC operator chooses an r_{max} of 100 km, the APM CSCI will continue loss calculations to 100 km, keeping the terrain profile flat from 95 km to 100 km with an elevation height of 50 m.

If *lerr6* is set to '.true.' and the furthest range point is less than r_{max}, then an error is returned in i_{error} from the APMINIT SU. This is to allow the TESS-NC CSCI application designer greater flexibility in how terrain data is handled.

3.6 SECURITY AND PRIVACY REQUIREMENTS

The security and privacy requirements are the same as those required by the target employing TESS-NC CSCI.

3.7 CSCI ENVIRONMENTAL REQUIREMENTS

The APM CSCI must be able to operate in the same hardware and software environments that the target employing TESS-NC CSCI operates.

3.8 COMPUTER RESOURCE REQUIREMENTS

Section 3.1.1.18 describes requirements for a Sine Fast Fourier Transform (SinFFT) SU. However, other sine FFT routines are available in the commercial market, and such a sine FFT may already be available within another TESS-NC CSCI. The selection of which FFT ultimately used by APM

mately used by APM CSCI is left to the application designer as every sine FFT will have hardware and/or software performance impacts.

3.9 SOFTWARE QUALITY FACTORS

The primary required quality factors can be divided into the three categories—design, performance, and adaptation.

The quality factors for the design category should include correctness, maintainability, and verifiability. Correctness describes the extent to which the APM CSCI conforms to its requirements and is to be determined from the criteria—completeness, consistency, and/or traceability. Maintainability specifies the effort required to locate and fix an error in the APM CSCI. Maintainability is to be determined from the criteria—consistency, modularity, self-descriptiveness (self-documentation), and/or simplicity. Verifiability characterizes the effort required to test the APM CSCI to ensure that it performs its intended function. Verifiability is to be determined from the criteria—modularity, self-descriptiveness, and/or simplicity.

The quality factor for performance category is reliability, which depicts the confidence that can be placed in the APM CSCI calculations. Reliability is to be determined from the criteria—accuracy, anomaly management, auditability, consistency, and/or simplicity.

The quality factors for the adaptation category are portability and reusability. Portability determines how easy it is to transport the APM CSCI from one hardware and/or software environment to another. Portability is to be determined from the criteria—application independence, modularity, and/or self-descriptiveness. Reusability illustrates how easy it is to convert the APM CSCI (or parts of the CSCI) for use in another application. Reusability is to be determined from the criteria - application independence, document accessibility, functional scope, generality, hardware independence, modularity, simplicity, self-descriptiveness, and/or system clarity.

Section A.1 defines the software quality criteria.

Only the software quality criteria completeness, consistency, and traceability can be analyzed. Their calculation is described in Section A.2. The other criteria have to be determined by either demonstration, test, or inspection.

3.10 DESIGN AND IMPLEMENTATION CONSTRAINTS

3.10.1 Implementation And Application Considerations

The calling TESS-NC CSCI application will determine the employment of the APM CSCI. However, the intensive computational nature of the APM CSCI must be taken into consideration when designing an efficient calling application. For this reason, the APM CSCI should be designed with flexibility for various hardware suites and computer resource management considerations. As stated in Section 1.1, this APM CSCI applies only to a coverage and loss diagram application. The following highly recommended guidelines are provided to aid in the design of a coverage or loss diagram application which will most efficiently employ the APM CSCI.

The APM CSCI propagation loss calculations are independent of any target or receiver considerations, therefore, for any EM emitter, one execution of the APM CSCI may be used to create both a coverage diagram and a loss diagram. Since both execution time and computer memory allocation

should be a consideration when employing this model, it is most efficient and appropriate to execute the APM CSCI for a particular EM system/environmental/terrain combination before executing any application. The output of the APM CSCI would be stored in a file which would be accessed by multiple applications.

For example, the TESS-NC operator may desire a coverage diagram for one particular radar system. At the beginning of the coverage diagram application, a check would be made for the existence of a previously created APM CSCI output file appropriate for the EM system, environmental, and terrain conditions. If such a file exists, the propagation loss values would be read from the file and used to create the coverage diagram. If the file does not exist, the APM CSCI would be executed to create one. As the APM CSCI is executing, its output could be routed simultaneously to a graphics display device and a file. This file could then be used in the loss diagram application should the operator also choose it. Two distinct applications, therefore, are achieved with only one execution of the APM CSCI. Additionally, should the operator desire an individual coverage diagram for each of multiple targets, or a single coverage diagram illustrating radar detection of a low-flying missile superimposed upon a coverage diagram illustrating his own radar's vulnerability as defined by the missile's ESM receiver, only a single execution of the APM CSCI would be required, thereby saving valuable computer resources.

3.10.2 Programming Language And Source Implementation

3.10.2.1 Programming Language. The ANSI Fortran 90 program language standard must be used in the development of the APM CSCI (reference h). This standard consists of the specifications of the language Fortran. With certain limitations the syntax and semantics of the old International Standard commonly known as "FORTRAN 77" are contained entirely within this new International Standard. Therefore, any standard-conforming FORTRAN 77 program is standard conforming under the Fortran 90 Standard. Note that the name of this language, Fortran, differs from that in FORTRAN 77 in that only the first letter is capitalized. The **Overview** section of the International Standard describes the major additions to FORTRAN 77 in this International Standard. Section 1.3 of the International Standard specifies the bounds of the Fortran language by identifying both those items included and those items excluded. Section 1.4.1 describes the FORTRAN 77 compatibility of the International Standard with emphasis on four FORTRAN 77 features having different interpolations in the new International Standard. The International Standard provides facilities that encourage the design and the use of modular and reusable software.

Section 8.2 of the International Standard describes nine obsolescent features of FORTRAN 77 that are redundant and for which better methods are available in FORTRAN 77 itself. These nine obsolescent features should not be used. These obsolescent features are:

1. **Arithmetic IF**—use the **IF** statement.

2. Real and double precision **DO** control variables and **DO** loop control expressions—use integer.

3. Shared **DO** termination and termination on a statement other than **END DO** or **CONTINUE**—use an **END DO** or a **CONTINUE** statement for each **DO** statement.

4. Branching to an **END IF** statement from outside its IF block—branch to the statement following the **END IF**.

5. Alternate return.
6. **PAUSE** statement.
7. **ASSIGN** and assigned **GO TO** statements.
8. Assigned **FORMAT** specifiers.
9. cH (nH) edit descriptor.

Remedies for the last five obsolescent features are described in section 8.2 of the International standard.

3.10.2.2 Source Implementation. Reference (f) by the Naval Oceanographic Office establishes a uniform standard for all software submitted by all contributors to them. It is recommended that the coding requirements set forth in Section 4 of that document be followed. Among these recommendations are:

1. Special non-ANSI features shall be avoided. Non-ANSI practices that are necessary must be documented in the code itself.
2. Maximum use should be made of existing commercially available FORTRAN callable libraries.
3. Programs shall be designed and coded using only five basic control structures - sequence of operations (assignment, add, ...), **IF THEN ELSE**, **DO WHILE**, **DO UNTIL**, and **CASE**.
4. Procedures or routines that make up a module shall not exceed an average of 100 executable statements per procedure or routine and shall not exceed a maximum of 200 executable statements in any procedure or routine.
5. Branching statements (**GO TO**s) shall only pass control to a statement that is in the same procedure or routine. Each **GO TO** must pass control only forward of its point of occurrence.
6. Naming conventions shall be uniform throughout the software. Program, subprogram, module, procedure, and data names shall be uniquely chosen to identify the applicable function performed. The naming convention for **COMMON** shall be consistent across the entire program.
7. Constants shall be defined not calculated (e.g., do no use *HALF = 1/2*, use *HALF = 0.5*)
8. Mixed-mode numerical operations should be avoided whenever possible. When determined to be necessary, the use shall be explicit (*FLOAT*, *FIX*, or in assignment statement) and completely described in comments.
9. Each component of the software shall have a prologue containing the name of the program, subprogram, or function and any version number; purpose; inputs; outputs; list of routines that call this routine; complete list of routines called including intrinsic functions such as *ABS* and *FLOAT*; glossary; and method.
10. To facilitate program comprehension, comment statements shall be used throughout the program code.

11. The use of the **EQUIVALENCE** statement shall be restricted to those where it either im proves the readability of the code or the efficiency of the program. If the **EQUIVALENCE** statement is used, it must be fully documented in the prologue and inline comment statements.
12. No machine-dependent techniques are allowed, unless there is no other way of perform ing the task.
13. Initialize every variable before use.
14. Do not depend on the values of "local" variables computed on a previous call to a routine.
15. Program structural indentation shall be used to improve readability and clarity.

3.11 PERSONNEL-RELATED REQUIREMENTS

Not applicable.

3.12 TRAINING RELATED REQUIREMENTS

The employing target software personnel implementing this CSCI into the TESS-NC CSCI will require training to become familiar with APM. This requirement should be met by this document and the companion Software Design Description (SDD) and Software Test Description (STD) documents.

3.13 OTHER REQUIREMENTS

None.

3.14 PRECEDENCE AND CRITICALITY OF REQUIREMENTS

The requirements presented in Sections 3.1 through 3.5 and Sections 3.8 through 3.10 have precedence over Sections 3.6, 3.7, 3.11, 3.12, and 3.13 and should be given equal weight.

4. QUALIFICATION PROVISIONS

N/A

5. REQUIREMENTS TRACEABILITY

5.1 SYSTEM TRACEABILITY

This section provides traceability of requirements between the APM CSCI and the TESS-NC CSCI.

1. The APM CSCI environmental data requirements should be obtained from the Tactical Environmental Data System database (TEDS) within the TESS-NC CSCI. The APM CSCI terrain data element requirements should be obtained from the Digital Terrain Elevation Database (DTED) within the TESS-NC CSCI. The radar/communication system

data element requirements should be obtained from the EM system database within the TESS-NC CSCI.

2. The TESS-NC CSCI requirement of propagation loss vs. range and height should be obtained from the APM CSCI.

5.2 DOCUMENTATION TRACEABILITY

This section provides the following types of traceability between the Software Requirements Specification (SRS), the Software Design Description (SDD), and the Software Test Description (STD):

1. Traceability between levels of requirements;
2. Traceability between the software requirements and software design;
3. Traceability between the software requirements and qualification test information obtained from the software testing.

This traceability of the Advanced Propagation Model is presented in two tables. The first table, table 7 given here, presents the traceability between levels of SRS requirements. The second table (table 101 in the Software Design Document) presents the traceability between the software requirements and software design.

Table 7. Requirements Traceability Matrix for the SRS.

Software Requirements Specification		Software Requirements Specification	
SRS Requirement Name	SRS Paragraph number	SRS Requirement Name	SRS Paragraph Number
CSCI Capability Requirements	3.1	Advance Propagation Initialization (APMINIT) CSC	3.1.1
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Allocate Arrays APM (ALLARRAY_APM) SU	3.1.1.1
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Allocate Array PE (ALLARRAY_PE) SU	3.1.1.2
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Allocate Array XO (ALLARRAY_XO) SU	3.1.1.3
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Dielectric Initialization (DIEINIT) SU	3.1.1.5
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Fast-Fourier Transform (FFT) SU	3.1.1.6
Fast-Fourier Transform (FFT) SU	3.1.1.6	Sine Fast-Fourier Transform (SINFFT) SU	3.1.1.18
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Fill Height Arrays (FILLHT) SU	3.1.1.8
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Gaseous Absorption (GASABS) SU	3.1.1.9
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Get Alpha Impedance (GETALN) SU	3.1.1.10
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Get Mode (GETMODE) SU	3.1.1.11
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Get Maximum Angle (GETTHMAX) SU	3.1.1.12
Get Maximum Angle (GETTHMAX) SU	3.1.1.12	FFT Parameters (FFTPAR) SU	3.1.1.7
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Interpolate Profile (INTPROF) SU	3.1.1.13
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Free-Space Propagator Phase Term (PHASE1) SU	3.1.1.14

Table 7. Requirements Traceability Matrix for the SRS. (Continued)

Software Requirements Specification		Software Requirements Specification	
SRS Requirement Name	SRS Paragraph number	SRS Requirement Name	SRS Paragraph Number
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Environmental Propagator Phase Term (PHASE2) SU	3.1.1.15
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Refractivity Initialization (REFINIT) SU	3.1.1.17
Refractivity Initialization (REFINIT) SU	3.1.1.17	Profile Reference (PROFREF) SU	3.1.1.16
Refractivity Initialization (REFINIT) SU	3.1.1.17	Remove Duplicate Refractivity Levels (REMDUP) SU	3.1.2.11
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Terrain Initialization (TERINIT) SU	3.1.1.19
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Troposcatter Initialization (TROPOINIT) SU	3.1.1.21
Troposcatter Initialization (TROPOINIT) SU	3.1.1.21	Antenna Pattern (ANTPAT) SU	3.1.1.4
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Starter Field Initialization (XYINIT) SU	3.1.1.22
CSCI Capability Requirements	3.1	Advance Propagation Model Step (APMSTEP) CSC	3.1.2
Advance Propagation Model Step (APMSTEP) CSC	3.1.2	Flat Earth Model (FEM) SU	3.1.2.3
Flat Earth Model (FEM) SU	3.1.2.3	Antenna Pattern (ANTPAT) SU	3.1.1.4
Flat Earth Model (FEM) SU	3.1.2.3	Get Reflection Coefficient (GETREFCOEF) SU	3.1.2.7
Advance Propagation Model Step (APMSTEP) CSC	3.1.2	Parabolic Equation Step (PESTEP) SU	3.1.2.8
Parabolic Equation Step (PESTEP) SU	3.1.2.8	Calculate Propagation Loss (CALCLOS) SU	3.1.2.1
Calculate Propagation Loss (CALCLOS) SU	3.1.2.1	Get Propagation Factor (GETPFAC) SU	3.1.2.6
Calculate Propagation Loss (CALCLOS) SU	3.1.2.1	Troposcatter (TROPO) SU	3.1.2.17

Table 7. Requirements Traceability Matrix for the SRS. (Continued)

Software Requirements Specification		Software Requirements Specification	
SRS Requirement Name	SRS Paragraph number	SRS Requirement Name	SRS Paragraph Number
Troposcatter (TROPO) SU	3.1.2.17	Antenna Pattern (ANTPAT) SU	3.1.1.4
Parabolic Equation Step (PESTEP) SU	3.1.2.8	DOSHIFT SU	3.1.2.2
Parabolic Equation Step (PESTEP) SU	3.1.2.8	Free Space Range Step (FRSTP) SU	3.1.2.4
Free Space Range Step (FRSTP) CSC	3.1.2.4	Fast-Fourier Transform (FFT) SU	3.1.1.6
Fast-Fourier Transform (FFT) SU	3.1.1.6	Sine Fast-Fourier Transform (SINFFT) SU	3.1.1.18
Parabolic Equation Step (PESTEP) SU	3.1.2.8	FZLIM SU	3.1.2.5
FZLIM SU	3.1.2.5	Get Propagation Factor (GETPFAC) SU	3.1.2.6
FZLIM SU	3.1.2.5	Save Profile (SAVEPRO) SU	3.1.2.15
FZLIM SU	3.1.2.5	Spectral Estimation (SPECEST) SU	3.1.2.16
Spectral Estimation (SPECEST) SU	3.1.2.16	Sine Fast-Fourier Transform (SINFFT) SU	3.1.1.18
Parabolic Equation Step (PESTEP) SU	3.1.2.8	Get Alpha Impedance (GETALN) SU	3.1.1.10
Parabolic Equation Step (PESTEP) SU	3.1.2.8	Refractivity Interpolation (REFINTER) SU	3.1.2.10
Refractivity Interpolation (REFINTER) SU	3.1.2.10	Interpolate Profile (INTPROF) SU	3.1.1.13
Refractivity Interpolation (REFINTER) SU	3.1.2.10	Profile Reference (PROFREF) SU	3.1.1.16
Refractivity Interpolation (REFINTER) SU	3.1.2.10	Remove Duplicate Refractivity Levels (REMDUP) SU	3.1.2.11
Parabolic Equation Step (PESTEP) SU	3.1.2.8	Environmental Propagator Phase Term (PHASE2) SU	3.1.1.15
Advance Propagation Model Step (APMSTEP) CSC	3.1.2	Ray Optics Model (ROM) SU	3.1.2.14

Table 7. Requirements Traceability Matrix for the SRS. (Continued)

Software Requirements Specification		Software Requirements Specification	
SRS Requirement Name	SRS Paragraph number	SRS Requirement Name	SRS Paragraph Number
Ray Optics Model (ROM) SU	3.1.2.14	Ray Optics Calculation (ROCALC) SU	3.1.2.12
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Antenna Pattern (ANTPAT) SU	3.1.1.4
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Get Reflection Coefficient (GETREFCOEF) SU	3.1.2.7
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Ray Trace (RAYTRACE) SU	3.1.2.9
Ray Optics Model (ROM) SU	3.1.2.14	Ray Optics Loss (ROLOSS) SU	3.1.2.13
CSCI Capability Requirements	3.1	Extended Optics Initialization (XOINIT) CSC	3.1.3
Extended Optics Initialization (XOINIT) CSC	3.1.3	Smooth (SMOOTH) SU	3.1.3.1
CSCI Capability Requirements	3.1	Extended Optics Step (XOSTEP) CSC	3.1.4
Extended Optics Step (XOSTEP) CSC	3.1.4	Extended Optics (EXTO) SU	3.1.4.1
Extended Optics (EXTO) SU	3.1.4.1	Troposcatter (TROPO) SU	3.1.2.17
Troposcatter (TROPO) SU	3.1.2.17	Antenna Pattern (ANTPAT) SU	3.1.1.4
Extended Optics Step (XOSTEP) CSC	3.1.4	Flat Earth Model (FEM) SU	3.1.2.3
Flat Earth Model (FEM) SU	3.1.2.3	Antenna Pattern (ANTPAT) SU	3.1.1.4
Flat Earth Model (FEM) SU	3.1.2.3	Get Reflection Coefficient (GETREFCOEF) SU	3.1.2.7
Extended Optics Step (XOSTEP) CSC	3.1.4	Ray Optics Model (ROM) SU	3.1.2.14

Table 7. Requirements Traceability Matrix for the SRS. (Continued)

Software Requirements Specification		Software Requirements Specification	
SRS Requirement Name	SRS Paragraph number	SRS Requirement Name	SRS Paragraph Number
Ray Optics Model (ROM) SU	3.1.2.14	Ray Optics Calculation (ROCALC) SU	3.1.2.12
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Antenna Pattern (ANTPAT) SU	3.1.1.4
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Get Reflection Coefficient (GETREFCOEF) SU	3.1.2.7
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Ray Trace (RAYTRACE) SU	3.1.2.9
Ray Optics Model (ROM) SU	3.1.2.14	Ray Optics Loss (ROLOSS) SU	3.1.2.13
CSCI Capability Requirements	3.1	Advanced Propagation Model Clean (APMCLEAN) CSC	3.1.5
CSCI Capability Requirements	3.1	CSCI External Interface Requirements	3.2
CSCI Capability Requirements	3.1	CSCI Internal Interface Requirements	3.3
CSCI Capability Requirements	3.1	CSCI Internal Data Requirements	3.4
CSCI Capability Requirements	3.1	Adaptation Requirements	3.5
CSCI Capability Requirements	3.1	Computer Resource Requirements	3.8
CSCI Capability Requirements	3.1	Software Quality Factors	3.9
CSCI Capability Requirements	3.1	Design And Implementation Constraints	3.10
Design And Implementation Constraints	3.10	Implementation and Application Considerations	3.10.1
Design And Implementation Constraints	3.10	Programming Language And Source Code Implementation	3.10.2

Table 7. Requirements Traceability Matrix for the SRS. (Continued)

Software Requirements Specification		Software Requirements Specification	
SRS Requirement Name	SRS Paragraph number	SRS Requirement Name	SRS Paragraph Number
Programming Language And Source Code Implementation	3.10.2	Programming Language	3.10.2.1
Programming Language And Source Code Implementation	3.10.2	Source Implementation	3.10.2.2
CSCI Capability Requirements	3.1	Personnel-Related Requirements	3.11
CSCI Capability Requirements	3.1	Training Related Requirements	3.12
CSCI Capability Requirements	3.1	Other Requirements	3.13
CSCI Capability Requirements	3.1	Precedence and Criticality of Requirements	3.14

6. NOTES

Table 8 is a glossary of acronyms and abbreviations used within this document. Table 9 is a glossary of Fortran terms used within this document.

Table 8. Acronyms and abbreviations.

Term	Definition
ANSI	American National Standards Institute
APM	Advanced Propagation Model
cB	centibel
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
dB	decibel
EM	Electromagnetic
FFT	Fast-Fourier Transform
Fortran	Formula Translation
FORTRAN	Formula Translation
km	kilometers

Table 8. Acronyms and abbreviations. (Continued)

Term	Definition
m	Meters
M	Modified refractivity units
MHz	Megahertz
N/A	Not applicable
PE	Parabolic Equation
p-space	Phase space
rad	Radians
SDD	Software Design Description
SRS	Software Requirements Specification
STD	Software Test Description
SU	Software Unit
TESS-NC	Tactical Environmental Support System Next Century
z-space	Distance space

Table 9. Fortran terms.

Term	Action or Definitions
ABS	Absolute value function
Arithmetic IF	Transfers control to one of three statement labels, depending on the value of *expression*
ASSIGN	Assigns the value of a format or statement label to an integer variable
CASE	Marks the beginning of a block of statements executed if an item in a list of expressions matches the test expressions
COMMON	Allows two or more program units to directly share variables without having to pass them as arguments
CONTINUE	Does not have any effect
DO	Repeatedly executes the statements following the **DO** statement through the statement which marks the end of the loop

Table 9. Fortran terms. (Continued)

Term	Action or Definitions
DO WHILE	Executes a block of statements repeatedly while a logical condition remains true
END DO	Terminates a **DO** or **DO WHILE** loop
END IF	Terminates a block of **IF** statements
EQUIVALENCE	Causes two or more variables or arrays to occupy the same memory location
FIX	Data type conversion function
FLOAT	Data type conversion function
FORMAT	Sets the format in which data is written to or read from a file
GO TO	Transfers execution to the statement label assigned to variable
IF	If expression is true, statement is executed; if expression is false, program execution continues weith the next executable statement
IF THEN ELSE	If expression is true, statements in the **IF** block are executed; if expression is false, control is transferred to the next **ELSE**, **ELSE IF**, or **END IF** statement at the same IF level
PAUSE	Temporarily suspends program execution and allows you to execute operating system commands during the suspension

APPENDIX A

A.1 DEFINITIONS OF QUALITY FACTOR CRITERIA

The criteria for judging the quality factors of Section 3.9 have the following definitions:

1. Accuracy. The precision of computations and control;
2. Anomaly management. The degree to which the program detects failure in order to maintain consistency;
3. Application independence. The degree to which the program is independent of nonstandard programming language features, operating system characteristics, and other environmental constraints;
4. Auditability. The ease with which conformance to standards can be checked;
5. Completeness. The degree to which full implementation of required function has been achieved;
6. Consistency. The use of uniform design and documentation techniques throughout the software development project;
7. Document accessibility. The availability of documents describing the program components.
8. Functional scope. The generality of the feature set and capabilities of the program;
9. Generality. The breadth of potential application of program components;
10. Hardware independence. The degree to which the software is decoupled from the hardware on which it operates;
11. Modularity. The functional independence of program components;
12. Self- descriptiveness. The degree to which the source code provides meaningful documentation;
13. Simplicity. The degree to which a program can be understood without difficulty;
14. System clarity. The ease for which the feature set and capabilities of the system can be determined.
15. Traceability. The ability to trace a design representation or actual program component back to requirements.

A.2 SOFTWARE QUALITY METRICS

A.2.1 Completeness Criteria

The criteria completeness can be determined from the metric:

1. no ambiguous references (input, function, output);
2. all data references defined;
3. all referenced functions defined;
4. all defined functions used;
5. all conditions and processing defined for each decision point;
6. all defined and referenced calling sequences parameters agree;
7. all problem reports resolved;
8. design agrees with requirements;
9. code agrees with design;
10. (score 0 for any untrue statement; 1 otherwise); and
11. metric value = SUM (scores)/9.

A.2.2 Consistency Criteria

The criteria consistency can be determined from the metric : number of modules violating the design standard divided by the number of modules.

A.2.3 Traceability Criteria

The criteria traceability can be determined from the metric: number of itemized requirements traced divided by the total number of requirements.

SOFTWARE DESIGN DESCRIPTION

FOR THE

ADVANCED PROPAGATION MODEL CSCI
(Version 1.0)

August 1998

Prepared for:

Space and Naval Warfare Systems Command (PMW-185)
San Diego, CA

Prepared by:

Space and Naval Warfare Systems Center, San Diego
Tropospheric Branch (Code D883)
49170 Propagation Path
San Diego, CA 92152-7385

CONTENTS

CONTENTS (CONTINUED)

CONTENTS (CONTINUED)

CONTENTS (CONITINUED)

Figures

Tables

Tables (Continued)

Tables (Continued)

1. SCOPE

1.1 IDENTIFICATION

The Advanced Propagation Model (APM) Version 1.0 computer software configuration item (CSCI) calculates range-dependent electromagnetic (EM) system propagation loss within a heterogeneous atmospheric medium over variable terrain, where the radio-frequency index of refraction is allowed to vary both vertically and horizontally, also accounting for terrain effects along the propagation path.

1.2 SYSTEM OVERVIEW

Numerous Tactical Environmental Support System–Next Century (TESS–NC) applications require EM-system propagation loss values. The APM model described in this document may be applied to two such TESS–NC applications, one that displays propagation loss on a range versus height scale (commonly referred to as a coverage diagram) and one that displays propagation loss on a propagation loss versus range/height scale (commonly referred to as a loss diagram).

1.3 DOCUMENT OVERVIEW

This document describes APM CSCI design and provides an input software requirement overview, a CSCI design architecture overview, and a detailed design description of each CSCI component.

2. REFERENCED DOCUMENTS

(a) Bergland, G. D. 1968. "A Radix-Eight Fast Fourier Transform Subroutine for Real-Valued Series," *IEEE Trans. Audio and Electro-Acoust.*, vol. AU-17, pp. 138-144.

(b) Cooley, J. W., P. A. W. Lewis, and P. D. Welsh. 1970. "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculation of Sine, Cosine and Laplace Transforms," *J. Sound Vib.*, vol. 12, pp. 315-337.

(c) Tappert, F. D. 1977. "The Parabolic Approximation Method." In *Wave Propagation and Underwater Acoustics*.pp. 224–285, J. B. Keller and J. S. Papadakis, Eds., Springer-Verlag, New York, NY.

(d) International Radio Consulting Committee (CCIR) XVth Plenary Assembly Dubrovnik. 1986. "Propagation in Non-Ionized Media," *Recommendations and Reports of the CCIR, 1986*, vol. V, International Telecommunications Union, Geneva, Switzerland..

(e) Commander-In-Chief, Pacific Fleet Meteorological Requirement (PAC MET) 87-04. 1987. "Range Dependent Electromagnetic Propagation Models."

(f) Dockery, G. D. 1988. "Modeling Electromagnetic Wave Propagation in the Troposphere Using the Parabolic Equation,"*IEEE Trans. Antennas Propagat.*, vol. 36, no. 10 (Oct), pp. 1464–1470.

(g) Naval Oceanographic Office. 1990. "Software Documentation Standards and Coding Requirements for Environmental System Product Development."

(h) Kuttler, J. R. and G. D. Dockery. 1991. "Theoretical Description of the Parabolic Approximation/Fourier Split-Step Method of Representing Electromagnetic Propagation in the Troposphere," *Radio Sci.,* vol. 26, pp. 381-393.

(i) American National Standards Institute (ANSI). 1992. "Program Language–Fortran–Extended." 1992.

(j) Patterson, W. L. and H. V. Hitney. 1992. "Radio Physical Optics CSCI Software Documents." NraD TD2403 (Dec), Naval Command, Control and Ocean Surveillance Center RDT&E Division*, San Diego, CA.

(k) Barrios, A. E. 1993. "Terrain and Refractivity Effects on Non-Optical Paths," *AGARD Conference Proceedings 543, Multiple Mechanism Propagation Paths (MMPPs): Their Characteristics and Influence on System Design*, Oct., pp. 10-1 to 10-9.

(l) Barrios, A. E. 1994. "A Terrain Parabolic Equation Model for Propagation in the Troposphere," *IEEE Trans. Antennas Propagat.,* vol. 42 (Jan), pp. 90-98.

(m) Naval Oceanographic Office. 1996. "Software Documentation Standards for Environmental System Product Development." Feb.

(n) Barrios, A. E. 1996. "Terrain Parabolic Equation Model (TPEM) Version 1.5 User's Manual." NraD TD 2898 (Feb), Naval Command, Control and Ocean Surveillance Center, RDT&E Division*, San Diego, CA.

(o) Sailors, D. B. and Barrios, A. E. 1997. "Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI) Documents." NRaD TD 2963 (May), Naval Command, Control and Ocean Surveillance Center, RDT&E Division*, San Diego, CA.

3. CSCI-WIDE DESIGN DECISIONS

The required APM CSCI propagation model is a range-dependent, true hybrid model that uses the complimentary strengths of both ray optics (RO) and parabolic equation (PE) techniques to calculate propagation loss both in range and altitude.

The atmospheric volume is divided into regions that lend themselves to the application of the various propagation loss calculation methods. Figure 1 illustrates these regions.

For antenna elevation angles above 5 degrees or for ranges less than approximately 2.5 km, a flat earth (FE), ray-optics model is used. In this region, only receiver height is corrected for average refraction and earth curvature.

Within the RO region (as defined by a limiting ray), propagation loss is calculated from the mutual interference between the direct-path and surface-reflected ray components using the refractivity profile at zero range. Full account is given to focusing or de-focusing along both direct and reflected ray paths and to the integrated optical path length difference between the two ray paths, to give precise phase difference, and, hence, accurate coherent sums for the computation of propagation loss.

---

* Now Space and Naval Warfare (SPAWAR) Systems Center, San Diego (SSC San Diego)

For the low-altitude region beyond the RO region, a PE approximation to the Helmholtz full-wave equation is employed. The PE model allows for range-dependent refractivity profiles and variable terrain along the propagation path and uses a split-step Fourier method for the PE solution. The PE model is run in the minimum region required to contain all terrain and trapping layer heights.

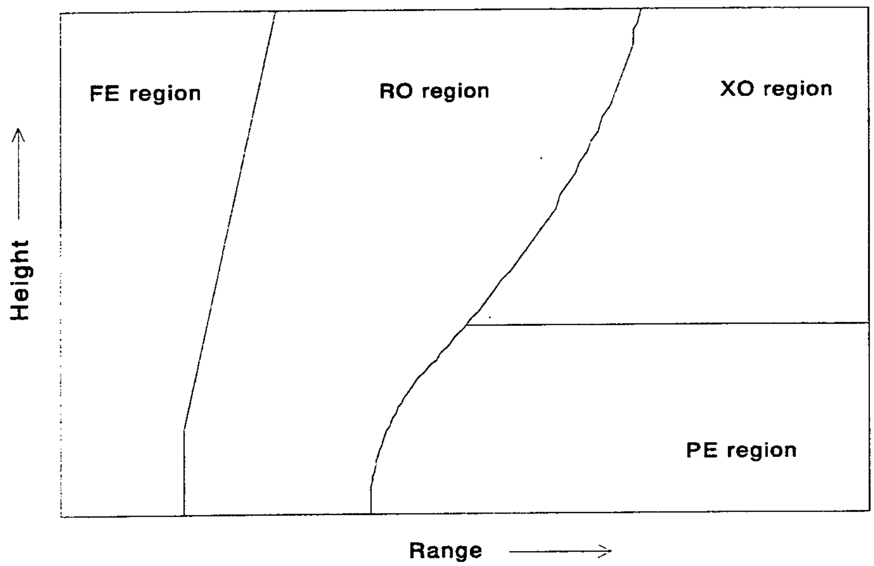

Figure 1. APM calculation regions.

For the area beyond the RO region, but above the PE region, an extended optics region (XO) is defined. Within the XO region, ray-optics methods that are initialized by the PE solution from below are used.

APM will run in three "execution" modes, depending on environmental inputs. APM will use the FE, RO, XO, and PE models if the terrain profile is flat for the first 2.5 km and if the antenna height is less than or equal to 100 m. It will use only the XO and PE models if the terrain profile is *not* flat for the first 2.5 km and if the antenna height is less than or equal to 100 m. APM will use only the PE model if the antenna height is greater than 100 m, regardless of terrain profile.

The APM CSCI allows for horizontal and vertical antenna polarization, finite conductivity based on user-specified ground composition and dielectric parameters, and the complete range of EM system parameters and most antenna patterns required by TESS–NC. APM also allows for gaseous absorption effects in all submodels and computes troposcatter losses within the diffraction region and beyond.

The APM CSCI is divided into five main computer software components (CSCs) and 40 additional software units (SUs). The first CSC, the APMINIT CSC, interfaces with various SUs for the complete initialization of the APM CSCI. The second CSC, the APMSTEP CSC, advances the entire APM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range. The XOINIT CSC initializes the range, height, and angle arrays in preparation for the XOSTEP CSC. The fourth CSC, the XOSTEP CSC, advances the APM CSCI algorithm one output step from the top of the PE calculation region to the maximum output height

specified, referencing various SUs to calculate the propagation output range. The last CSC, the APMCLEAN CSC, deallocates all dynamically dimensioned arrays in one complete run of APM calculations.

4. CSCI ARCHITECTURE DESIGN

4.1 CSCI COMPONENTS

The APM CSCI is accessed by a subroutine call that provides, as global data elements, the values specified in tables 1 through 4.

The APM CSCI is divided into five CSCs and 40 SUs. The five CSCs are the APMINIT CSC, the APMSTEP CSC, the XOINIT CSC, the XOSTEP CSC, and the APMCLEAN CSC. The source code for the APM CSCI is listed in Appendix A. The name and purpose for each CSC and SU are listed below.

The Advance Propagation Initialization (APMINIT) CSC interfaces with various SUs for the complete initialization of the APM CSCI. The APMINIT CSC component SUs include the following:

1. **Allocate Arrays APM (ALLARRAY_APM) SU.** Allocates and initializes all dynamically dimensioned arrays associated with APM terrain, refractivity, troposcatter, and general variable arrays.
2. **Allocate Array PE (ALLARRAY_PE) SU.** Allocates and initializes all dynamically dimensioned arrays associated with PE calculations.
3. **Allocate Array XO (ALLARRAY_XO) SU**. Allocates and initializes all dynamically dimensioned arrays associated with XO calculations.
4. **Antenna Pattern (ANTPAT) SU.** Calculates a normalized antenna gain (antenna pattern factor) for a specified antenna elevation angle.
5. **Dielectric Initialization (DIEINIT) SU.** Determines the conductivity and relative permittivity as a function of frequency in MHz based on general ground composition types.
6. **Fast-Fourier Transform (FFT) SU.** Separates the real and imaginary components of the complex PE field into two real arrays and then references the SINFFT SU.
7. **FFT Parameters (FFTPAR) SU.** Determines the required transform size based on the maximum PE propagation angle and the maximum height needed.
8. **Fill Height Arrays (FILLHT) SU.** Calculates the effective earth radius for an initial launch angle of 5 degrees and fills an array with height values at each output range of the limiting sub-model, depending on which mode is used.
9. **Gaseous Absorption (GASABS) SU.** Computes the specific attenuation based on air temperature and absolute humidity.
10. **Get Alpha Impedance (GETALN) SU.** Computes the impedance term in the Leontovich boundary condition, and the complex index of refraction for finite conductivity and vertical polarization calculations.
11. **Get Mode (GETMODE) SU.** Determines what "execution" mode APM will run based on environmental inputs for the current application.

12. **Get Maximum Angle (GETTHMAX) SU.** Performs an iterative ray trace to determine the minimum angle required (based on the reflected ray) in obtaining a PE solution.
13. **Interpolate Profile (INTPROF) SU.** Performs a linear interpolation vertically with height on the refractivity profile.
14. **Free Space Propagator Phase Term (PHASE1) SU.** Initializes the free space propagator array for subsequent use in the PESTEP SU.
15. **Environmental Propagator Phase Term (PHASE2) SU.** Calculates the environmental phase term for an interpolated environment profile.
16. **Profile Reference (PROFREF) SU.** Adjusts the current refractivity profile so that it is relative to a reference height.
17. **Refractivity Initialization (REFINIT) SU.** Checks for valid environmental profile inputs and initializes the refractivity arrays.
18. **Sine Fast Fourier Transform (SINFFT) SU.** Transforms each portion of the PE solution.
19. **Terrain Initialization (TERINIT) SU.** Examines and initializes terrain arrays for subsequent use in PE calculations.
20. **Troposcatter Initialization (TROPOINIT) SU.** Initializes all variables and arrays needed for subsequent troposcatter calculations.
21. **Starter Field Initialization (XYINIT) SU.** Calculates the complex PE solution at range zero.

The Advanced Propagation Model Step (APMSTEP) CSC advances the entire APM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range. The APMSTEP CSC component SUs include the following:

1. **Calculate Propagation Loss (CALCLOS) SU.** Determines the propagation loss from the complex PE field at each output height point at the current output range.
2. **DOSHIFT SU.** Shifts the field by the number of bins or PE mesh heights corresponding to local ground height.
3. **Flat Earth Model (FEM) SU.** Computes propagation loss at a specified range based upon flat-earth approximations.
4. **Free Space Range Step (FRSTP) SU.** Propagates the complex PE solution field in free space by one range step.
5. **FZLIM SU.** Determines both the propagation factor (in dB) and the outgoing propagation angle at the top of the PE calculation region.
6. **Get Propagation Factor (GETPFAC) SU.** Determines the propagation factor at the specified height in dB.
7. **Get Reflection Coefficient (GETREFCOEF) SU.** Calculates the complex surface reflection coefficient, along with the magnitude and phase angle.
8. **Parabolic Equation Step (PESTEP) SU.** Determines the next output range and begins an iterative loop to advance the PE solution such that for the current PE range, a PE solution is calculated from the solution at the previous PE range. This procedure is repeated until the output range is reached.

9. **Ray Trace (RAYTRACE) SU.** Traces a ray from a starting height and range with a specified starting elevation angle to a termination range.
10. **Refractivity Interpolation (REFINTER) SU.** Interpolates both horizontally and vertically on the modified refractivity profiles.
11. **Remove Duplicate Refractivity Levels (REMDUP) SU.** Removes any duplicate refractivity levels in the currently interpolated profile.
12. **Ray Optics Calculation (ROCALC SU).** Computes the RO components that will be needed in the calculation of propagation loss at a specified range and height within the RO region.
13. **Ray Optics Loss (ROLOSS) SU.** Calculates both the propagation loss values at a specified range and height based upon the components of magnitude for a direct-path and surface-reflected ray and the total phase lag angle between the two rays as determined by the ROCALC SU.
14. **Ray Optics Model (ROM) SU.** Provides a one-call routine for RO calculations.
15. **Save Profile (SAVEPRO) SU.** Stores the refractivity profiles at each PE range step from the top of the PE region to the maximum user-specified height.
16. **Spectral Estimation (SPECEST) SU.** Determines, via spectral estimation, the outward propagation angle at the top of the PE calculation region.
17. **Troposcatter (TROPO) SU.** Determines the loss due to troposcatter and computes the appropriate loss between troposcatter loss and propagation loss in the "transition" region using a method of "bold interpolation."

The Extended Optics Initialization (XOINIT) CSC initializes the range, height, and angle arrays in preparation for the XOSTEP CSC. The XOINIT CSC component SUs include:

Smooth (SMOOTH) SU. Performs an n-point average smoothing on any array passed to it.

The Extended Optics Step (XOSTEP) CSC advances the APM CSCI algorithm one output range step from the top of the PE calculation region to the maximum output height specified, referencing various SUs to calculate the propagation loss at the current output range. The XOSTEP CSC component SUs include:

Extended Optics (EXTO) SU. Calculates propagation loss, based on extended optics techniques, at the current output range.

The Advanced Propagation Model Clean (APMCLEAN) CSC deallocates all dynamically dimensioned arrays used in one complete run of APM calculations.

4.2 CONCEPT OF EXECUTION

Figure 2 shows the program flow of the required APM CSCI. Note that the APM CSCI is shown within the context of a calling CSCI application such as one that generates a coverage or loss diagram. The efficient implementation of the APM CSCI will have far-reaching consequences upon the design of an application CSCI beyond those mentioned in section 7.3. For example, figure 2 shows checking for the existence of a previously created APM output file prior to the access of the APM CSCI. The application CSCI will have to consider if the atmospheric or terrain environment has changed since the APM output file was created or if any new height or range requirement is accom-

modated within the existing APM CSCI output file. Because these and many more considerations are beyond the scope of this document, an application CSCI designer should work closely with the APM CSCI development agency in APM CSCI implementation.

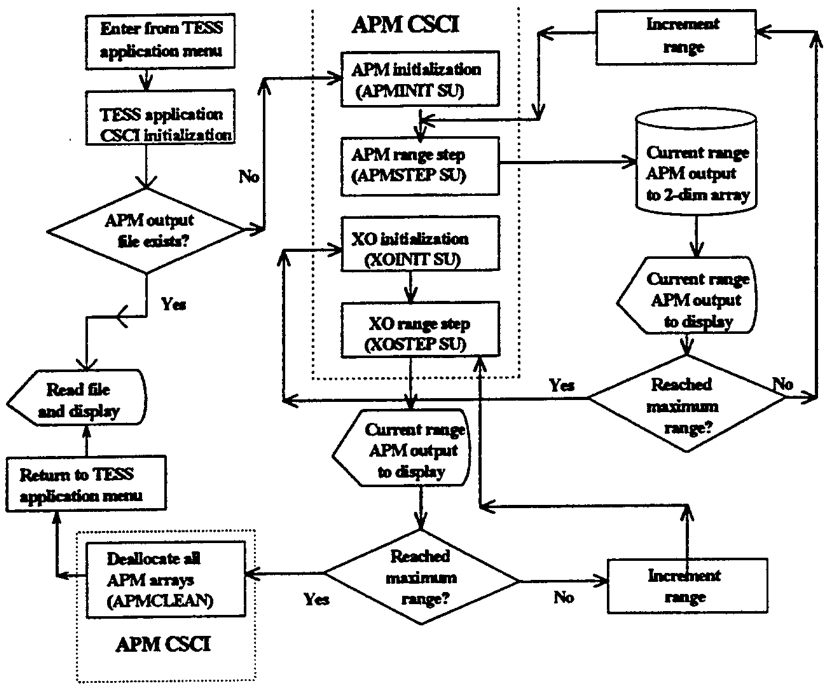

Figure 2. APM CSCI program flow.

4.3 INTERFACE DESIGN

4.3.1 Interface Identification and Diagrams

The APM CSCI interface design consists of one FORTRAN MODULE file for the external and internal data interface, FORTRAN CALL statements for both output data and internal interfacing, and several FORTRAN COMMON blocks for the internal interface. The MODULE file is called APM_MOD. This MODULE's statements provide several constants, the COMMON blocks, and the dynamically allocated array names. The COMMON block names are: 1) ABSORB, 2) ERRORFLAG, 3) IMPEDANCE, 4) INPUTVAR, 5) MISCVAR, 6) OUTRH, 7) PATTERN, 8) PE, 9) REFPROF, 10) REFRACTIVITY, 11) RO, 12) SPEC, 13) SYSTEMVAR, 14) TERRAIN, 15) TROPOV, 16) TRVAR, and 17) XO.

4.3.2 External Interface

The APM CSCI is accessed through the APMINIT CSC by a subroutine call from the TESS–NC CSCI, which should provide, as global data elements, the values specified in tables 1 through 4.

The APM CSCI external data elements (i.e., data that must be provided by the calling TESS–NC CSCI in the MODULE file prior to the APM CSCI execution) can be divided into four classifications. The first are external data related to the atmospheric environment, specified in table 1; the second are data related to the EM system, specified in table 2; the third are data related to the implementation of the APM CSCI by the TESS–NC CSCI, specified in table 3; and the fourth are data

calling CSCI via the FORTRAN CALL statements.

Table 1. APM CSCI environmental data element requirements.

Name	Description	Type	Units	Bounds
refmsl	Profile modified refractivity (dynamically allocated) array referenced to mean sea level	Real	M-units	≥ 0.0 [a]
hmsl	Profile height (dynamically allocated) array	Real	meters	≥ 0.0 [a]
n_{prof}	Number of profiles	Integer	N/A	≥ 1
lvlp	Number of profile levels	Integer	N/A	≥ 2
rngprof	Dynamically allocated array of ranges to each profile	Real	meters	≥ 0.0
abs_{hum}	Surface absolute humidity	Real	g/m^3	0 to 50
t_{air}	Surface air temperature	Real	°C	-20 to 40
γ_a	Surface specific attenuation	Real	dB/km	≥0.0
i_{extra}	Extrapolation flag for refractivity profiles entered below mean sea level	Integer	N/A	0 or 1

[a] Couplets of height and modified refractivity associated with that height are referred to within this document as an environmental profile.

Table 2. APM CSCI External EM System data element requirements.

Name	Description	Type	Units	Bounds
μ_{bw}	Antenna vertical beam width	Real	degree	0.5 to 45.0
μ_o	Antenna elevation angle	Real	degree	-50.0 to 50.0
f_{MHz}	EM system frequency	Real	MHz	100.0 to 20,000.0
i_{pat}	Antenna pattern 1 = Omni-directional 2 = Gaussian 3 = Sine (X)/X 4 = Cosecant-squared 5 = Generic height-finder 6 = User-defined height-finder	Integer	N/A	1 to 6
i_{pol}	Antenna polarization 0 = Horizontal 1 = Vertical	Integer	N/A	0 or 1

Table 2. APM CSCI External EM System data element requirements. (Continued)

Name	Description	Type	Units	Bounds
ant_{ht}	Antenna height above local ground at range 0.0 m	Real	meters	≥ 1.0
$hfang$	Dynamically allocated user-defined height-finder power-reduction angle array	Real	degree	0.0 to 90.0
$hffac$	Dynamically allocated user-defined power-reduction factor array	Real	N/A	0.0 to 1.0
n_{facs}	Number of power-reduction angles/factors for user-defined height-finder radar	Integer	N/A	1 to 10

Table 3. APM CSCI external implementation constants.

Name	Description	Type	Units	Bounds
n_{rout}	Number of range output points for a particular application of APM	Integer	N/A	1
n_{zout}	Number of height output points for a particular application of APM	Integer	N/A	1
$lerr6$	Logical flag to allow for error -6 to be bypassed	Logical	N/A	'.true.' or '.false.' [a]
$lerr12$	Logical flag to allow for error -12 to be bypassed	Logical	N/A	'.true.' or '.false.' [a]
i_{tropo}	Flag to include troposcatter calculations (0 = no, 1 = yes)	Integer	N/A	0 or 1
r_{max}	Maximum range output for a particular application of APM	Real	meters	≥ 5000.0 [b]
h_{min}	Minimum height output for a particular application of APM	Real	meters	≥ 0.0 [c]
h_{max}	Maximum height output for a particular application of APM	Real	meters	≥ 100.0 [b]

[a] refer to section 7.2 for a complete description.

[b] refer to section 7.3 for a complete description.

Table 4. APM CSCI external terrain data element requirements.

Name	Description	Type	Units	Bounds
terx	Dynamically allocated terrain profile range array	Real	meters	≥ 0.0 [a]
tery	Dynamically allocated terrain profile height array	Real	meters	≥ 0.0 [a]
i_{tp}	Number of terrain profile points for a particular application of APM	Integer	N/A	≥ 2
i_{gr}	Number of ground types for a particular application of APM	Integer	N/A	≥ 0.0 [a]
igrnd	Array of ground composition types for a particular application of APM 0 = Sea water 1 = Freshwater 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1°C 6 = Ice at -10°C 7 = User defined	Integer	N/A	$0 \leq igrnd_i \leq 7$ [a]
rgrnd	Dynamically allocated array of ranges for which ground types are applied for a particular application of APM	Real	meters	≥ 0.0 [a]
dielec	Dynamically allocated two-dimensional array of relative permittivity and conductivity for a particular application of APM	Real	N/A	>0 [a]

[a] Refer to section 7.3 for a complete description.

Table 5. APM CSCI output data element requirements.

Name	Description	Type	Units	Source
i_{xostp}	Index of output range step at which XO model is to be applied	Integer	N/A	APMINIT CSC
i_{error}	Integer value that is returned if an error occurs in called routine	Integer	N/A	APMINIT CSC XOINIT CSC APMCLEAN CSC

Table 5. APM CSCI output data element requirements. (Continued)

Name	Description	Type	Units	Source
mloss	Propagation loss	Integer	cB	APMSTEP CSC XOSTEP CSC
j_{start}	Output height index at which valid PE propagation loss values begin	Integer	N/A	APMSTEP CSC
j_{end}	Output height index at which valid PE propagation loss values end	Integer	N/A	APMSTEP CSC
r_{out}	Current range	Real	meters	APMSTEP CSC XOSTEP CSC
j_{xstart}	Output height index at which valid XO propagation loss values begin	Integer	N/A	XOINIT CSC
j_{xend}	Output height index at which valid XO propagation loss values end	Integer	N/A	XOSTEP CSC

[a]Refer to section 4.3.4 for a complete description.

4.3.3 Internal Interface

Section 4.2 shows the relationship between the APM CSCI and its five main CSCIs: APMINIT, AMPSTEP, XOINIT, XOSTEP, and APMCLEAN. Figure 1 shows this relationship. The internal interface between these three CSCs and the APM CSCI is left to the design. However, table 6 shows the internal structure of the APM CSCI and its CSCs and SUs. The two left columns show the calling subroutines, and the two right columns show the subroutines called. Columns 2 and 4 give the section number where more details about the various CSCs and SUs of the APM CSCI can be found.

Table 6. APM internal interface design.

Software Design Description		Software Design Description	
Software Design Description Name	SDD Section Number	Software Design Description Name	SDD Section Number
CSCI Detailed Design	5	Advance Propagation Initialization (APMINIT) CSC	5.1
Advance Propagation Initialization (APMINIT) CSC	5.1	Allocate Arrays APM (ALLARRAY_APM) SU	5.1.1
Advance Propagation Initialization (APMINIT) CSC	5.1	Allocate Array PE (ALLARRAY_PE) SU	5.1.2
Advance Propagation Initialization (APMINIT) CSC	5.1	Allocate Array XO (ALLARRAY_XO) SU	5.1.3
Advance Propagation Initialization (APMINIT) CSC	5.1	Dielectric Initialization (DIEINIT) SU	5.1.5
Advance Propagation Initialization (APMINIT) CSC	5.1	Fast Fourier Transform (FFT) SU	5.1.6

Table 6. APM internal interface design. (Continued)

Software Design Description		Software Design Description	
Fast Fourier Transform (FFT) SU	5.1.6	Sine Fast Fourier Transform (SINFFT) SU	5.1.18
Advance Propagation Initialization (APMINIT) CSC	5.1	Fill Height Arrays (FILLHT) SU	5.1.8
Advance Propagation Initialization (APMINIT) CSC	5.1	Gaseous Absorption (GASABS) SU	5.1.9
Advance Propagation Initialization (APMINIT) CSC	5.1	Get Alpha Impedance (GETALN) SU	5.1.10
Advance Propagation Initialization (APMINIT) CSC	5.1	Get Mode (GETMODE) SU	5.1.11
Advance Propagation Initialization (APMINIT) CSC	5.1	Get Maximum Angle (GETTHMAX) SU	5.1.12
Get Maximum Angle (GETTHMAX) SU	5.1.12	FFT Parameters (FFTPAR) SU	5.1.7
Advance Propagation Initialization (APMINIT) CSC	5.1	Interpolate Profile (INTPROF) SU	5.1.13
Advance Propagation Initialization (APMINIT) CSC	5.1	Free Space Propagator Phase Term (PHASE1) SU	5.1.14
Advance Propagation Initialization (APMINIT) CSC	5.1	Environmental Propagator Phase Term (PHASE2) SU	5.1.15
Advance Propagation Initialization (APMINIT) CSC	5.1	Refractivity Initialization (REFINIT) SU	5.1.17
Refractivity Initialization (REFINIT) SU	5.1.17	Profile Reference (PROFREF) SU	5.1.16
Refractivity Initialization (REFINIT) SU	5.1.17	Remove Duplicate Refractivity Levels (REMDUP) SU	5.1.11
Advance Propagation Initialization (APMINIT) CSC	5.1	Terrain Initialization (TERINIT) SU	5.1.19
Advance Propagation Initialization (APMINIT) CSC	5.1	Troposcatter Initialization (TROPOINIT) SU	5.1.20
Troposcatter Initialization (TROPOINIT) SU	5.1.20	Antenna Pattern (ANTPAT) SU	5.1.4
Advance Propagation Initialization (APMINIT) CSC	5.1	Starter Field Initialization (XYINIT) SU	5.1.21
CSCI Detailed Design	5.	Advance Propagation Model Step (APMSTEP) CSC	5.2
Advance Propagation Model Step (APMSTEP) CSC	5.2	Flat Earth Model (FEM) SU	5.2.3

Table 6. APM internal interface design. (Continued)

Software Design Description		Software Design Description	
Flat Earth Model (FEM) SU	5.2.3	Antenna Pattern (ANTPAT) SU	5.1.4
Flat Earth Model (FEM) SU	5.2.3	Get Reflection Coefficient (GETREFCOEF) SU	5.2.7
Advance Propagation Model Step (APMSTEP) CSC	5.2	Parabolic Equation Step (PESTEP) SU	5.2.8
Parabolic Equation Step (PESTEP) SU	5.2.8	Calculate Propagation Loss (CALCLOS) SU	5.2.1
Calculate Propagation Loss (CALCLOS) SU	5.2.1	Get Propagation Factor (GETPFAC) SU	5.2.6
Calculate Propagation Loss (CALCLOS) SU	5.2.1	Troposcatter (TROPO) SU	5.2.17
Troposcatter (TROPO) SU	5.2.17	Antenna Pattern (ANTPAT) SU	5.1.4
Parabolic Equation Step (PESTEP) SU	5.2.8	DoShift SU	5.2.2
Parabolic Equation Step (PESTEP) SU	5.2.8	Free Space Range Step (FRSTP) SU	5.2.4
Free Space Range Step (FRSTP) CSC	5.2.4	Fast Fourier Transform (FFT) SU	5.1.6
Fast Fourier Transform (FFT) SU	5.1.6	Sine Fast Fourier Transform (SINFFT) SU	5.1.18
Parabolic Equation Step (PESTEP) SU	5.2.8	FZLIM SU	5.2.5
FZLIM SU	5.2.5	Get Propagation Factor (GETPFAC) SU	5.2.6
FZLIM SU	5.2.5	Save Profile (SAVEPRO) SU	5.2.15
FZLIM SU	5.2.5	Spectral Estimation (SPECEST) SU	5.2.16
Spectral Estimation (SPECEST) SU	5.2.16	Sine Fast Fourier Transform (SINFFT) SU	5.1.18
Parabolic Equation Step (PESTEP) SU	5.2.8	Get Alpha Impedance (GETALN) SU	5.1.10
Parabolic Equation Step (PESTEP) SU	5.2.8	Refractivity Interpolation (REFINTER) SU	5.2.10
Refractivity Interpolation (REFINTER) SU	5.2.10	Interpolate Profile (INTPROF) SU	5.1.13
Refractivity Interpolation (REFINTER) SU	5.2.10	Profile Reference (PROFREF) SU	5.1.16
Refractivity Interpolation (REFINTER) SU	5.2.10	Remove Duplicate Refractivity Levels (REMDUP) SU	5.2.11

Table 6. APM internal interface design. (Continued)

Software Design Description		Software Design Description	
Parabolic Equation Step (PESTEP) SU	5.2.8	Environmental Propagator Phase Term (PHASE2) SU	5.1.15
Advance Propagation Model Step (APMSTEP) CSC	5.2	Ray Optics Model (ROM) SU	5.2.14
Ray Optics Model (ROM) SU	5.2.14	Ray Optics Calculation (ROCALC) SU	5.2.12
Ray Optics Calculation (ROCALC) SU	5.2.12	Antenna Pattern (ANTPAT) SU	5.1.4
Ray Optics Calculation (ROCALC) SU	5.2.12	Get Reflection Coefficient (GETREFCOEF) SU	5.2.7
Ray Optics Calculation (ROCALC) SU	5.2.12	Ray Trace (RAYTRACE) SU	5.2.9
Ray Optics Model (ROM) SU	5.2.14	Ray Optics Loss (ROLOSS) SU	5.2.13
CSCI Detailed Design	5.	Extended Optics Initialization (XOINIT) CSC	5.3
Extended Optics Initialization (XOINIT) CSC	5.3	Smooth (Smooth) su	5.3.1
CSCI Detailed Design	5	Extended Optics Step (XOSTEP) CSC	5.4
Extended Optics Step (XOSTEP) CSC	5.4	Extended Optics (EXTO) SU	5.4.1
Extended Optics (EXTO) SU	5.4.1	Troposcatter (TROPO) SU	5.2.17
Troposcatter (TROPO) SU	5.2.17	Antenna Pattern (ANTPAT) SU	5.1.4
Extended Optics Step (XOSTEP) CSC	5.4	Flat Earth Model (FEM) SU	5.2.3
Flat Earth Model (FEM) SU	5.2.3	Antenna Pattern (ANTPAT) SU	5.1.4
Flat Earth Model (FEM) SU	5.2.3	Get Reflection Coefficient (GETREFCOEF) SU	5.2.7
CSCI DETAILED DESIGN	5	Extended Optics Step (XOSTEP) CSC	5.4
Extended Optics Step (XOSTEP) CSC	5.4	Ray Optics Model (ROM) SU	5.2.14
Ray Optics Model (ROM) SU	5.2.14	Ray Optics Calculation (ROCALC) SU	5.2.12
Ray Optics Calculation (ROCALC) SU	5.2.12	Antenna Pattern (ANTPAT) SU	5.1.4
Ray Optics Calculation (ROCALC) SU	5.2.12	Get Reflection Coefficient (GETREFCOEF) SU	5.2.7
Ray Optics Calculation (ROCALC) SU	5.2.12	Ray Trace (RAYTRACE) SU	5.2.9
Ray Optics Model (ROM) SU	5.2.14	Ray Optics Loss (ROLOSS) SU	5.2.13
CSCI Detailed Design	5	Advanced Propagation Model Clean (APMCLEAN) CSC	5.5

4.3.4 Internal Data

The APM CSCI takes full advantage of Fortran 90 features, using allocatable arrays for all internal and input arrays. This utilization requires that the TESS–NC CSCI designer correctly allocate and initialize all arrays necessary for APM CSCI input. The APMCLEAN CSC is provided as part of the APM CSCI and can be called by the TESS–NC application to deallocate all arrays used by the APM CSCI in one complete run.

Due to the computational intensity of the APM CSCI, it may not be necessary or desirable to use the extreme capability of the APM CSCI for all applications. The variables n_{rout} and n_{zout} refer to the desired number of range and height output points for any one particular application, and will be specified when the APMINIT CSC is called.

One of the parameters returned to the TESS–NC application from the APMINIT CSC is i_{error}. Returning the parameter allows greater flexibility in how input data are handled within the TESS application. Table 7 lists all possible errors that can be returned.

The logical variables, *lerr6* and *lerr12*, when set to '.false.', allow the TESS–NC application to bypass their associated errors, as these are not critical to APM CSCI operation.

The APM CSCI provides propagation loss for all heights and ranges when running in a full hybrid mode. When running in a partial hybrid mode, it provides propagation loss for all heights, but not necessarily for all angles. Finally, it will be limited in both height and angle coverage when running in a PE-only mode. Refer to section 5 for environmental conditions under which each execution mode is automatically selected.

Absorption by atmospheric gases (oxygen and water vapor) may be important to some APM CSCI applications and is controlled by specifying a non-zero value for the absolute humidity, abs_{hum}, and the surface air temperature, t_{air}; or likewise, specifying a non-zero value for the gaseous absorption attenuation rate, γ_a.

Table 7. APMINIT SU returned error definitions.

Error	Definition
-6	Last range in terrain profile is less than r_{max}. Returns this error only if lerr6 is set to '.true.'
-7	Specified cut-back angles (for user-defined height finder antenna pattern) are not increasing.
-8	h_{max} is less than maximum height of terrain profile.
-9	Antenna height with respect to mean sea level is greater than maximum height h_{max}.
-10	Beamwidth is less than or equal to zero for directional antenna pattern.
-12	Range of last environment profile given (for range-dependent case) is less than r_{max}. Returns this error only if lerr12 is set to '.true.'
-13	Height of first level in any user-specified refractivity profile is greater than 0. First height must be at mean sea level (0.0) or < 0.0 if below mean sea level.
-14	Last gradient in any environment profile is negative.
-17	Range points of terrain profile are not increasing.
-18	First range value in terrain profile is not 0.
-42	Minimum height input by user, h_{min}, is greater than maximum height, h_{max}.

A particular APM CSCI application may or may not require the consideration of troposcatter effects within the propagation loss calculations. For example, a radar evaluation would, most likely, not be influenced by troposcatter while an ESM evaluation would. APM has the feature of including or not including the troposcatter calculation by setting a parameter called i_{tropo}. Setting this parameter to 0 omits the calculation. Setting this parameter to 1 includes the calculation. For the APM CSCI implementation within the TESS–NC coverage and loss diagram applications, i_{tropo} must be set equal to 1 to include the calculation.

5. CSCI DETAILED DESIGN

The following subsections provide a description of each APM CSCI component.

5.1 ADVANCE PROPAGATION INITIALIZATION (APMINIT) CSC

The APMINIT CSC interfaces with various SUs for the complete initialization of the APM CSCI.

Upon entering the APMINIT CSC, several variables are initialized. The error flag (i_{error}), the maximum PE propagation angle (Θ_{max}), the absorption calculation flag (k_{abs}), and the range at which PE loss values will start being calculated (r_{pest}), are set to 0.

Next, the absorption calculation flag, k_{abs}, is set to 1 if the air temperature, t_{air}, or the absolute humidity, abs_{hum}, are non-zero. If an attenuation rate is specified ($\gamma_a \neq 0$), then k_{abs} is set to 2.

Next, the following variables are checked for valid numerical values: maximum output range, r_{max}; maximum output height, h_{max}; and minimum output height, h_{min}. The variable r_{max}, is set to the value specified from the calling CSCI or 5000 meters, whichever is greater. The variable h_{max}, is set to the value specified from the calling CSCI or 100 meters, whichever is greater. If h_{min} is greater than h_{max}, then i_{error} is set to -42 and the APMINIT CSC is exited. If the maximum output range and minimum and maximum output height values are valid, then the APMINIT CSC proceeds to the next step.

The atmospheric volume must be "covered" or resolved with a mesh of calculation points that will, as a matter of routine, exceed the height/range resolution requirements of the particular application of the APM CSCI. The height and range mesh size per APM CSCI output point, Δz_{out} and Δr_{out}, respectively, are calculated from the number of APM output points and the maximum range and height as follows:

$$\Delta r_{out} = \frac{r_{max}}{n_{rout}},$$

$$\Delta z_{out} = \frac{h_{max} - h_{min}}{n_{zout}}.$$

The array, *zout*, which contains all output height points, is determined by the formula:

$$zout_i \;=\; h_{min} \;+\; i\,\Delta z_{out} \qquad for\ i = 1,2,3,\ldots n_{zout}.$$

Next, the following variables are determined for later calculations: the wavelength, λ; the free-space wavenumber, k_o; a multiplicative variable, *con*, used to determine the refractivity phase term; and a loss term used to determine free-space loss, pl_{cnst}. They are determined as follows:

$$\lambda = \frac{c_o}{f_{MHz}},$$

where c_o is the speed of light (299.79×10^6 m/s);

$$k_o = \frac{2\pi}{\lambda},$$

$$con = 10^6 \ k_o,$$

$$pl_{cnst} = 20\,\mathrm{LOG}(2k_o).$$

The number of terrain range/height pairs, i_{tpa}, used for internal calculations, is initialized to 1 plus the user-specified number of range/height pairs, i_{tp}. The ALLARRAY_APM SU is then referenced to dynamically allocate and initialize all arrays associated with terrain, refractivity, troposcatter, and general variable arrays. If an error has occurred while allocating memory, i_{error} is returned with a non-zero value and the CSC is exited; otherwise, the CSC proceeds to the next step.

Arrays containing: all output ranges, *rngout;* the square of all output ranges, *rsqrd;* 20 times the logarithm of all output ranges, *rlogo* and the free-space loss at all output ranges, *fslr,* are initialized as follows:

$$rngout_i = i\,\Delta r_{out},$$
$$rsqrd_i = (i\,\Delta r_{out})^2,$$
$$rlogo_i = 20\,\mathrm{LOG}(i\,\Delta r_{out}),$$
$$fslr_i = rlogo_i + pl_{cnst};$$

where the index i ranges from 1 to n_{rout}.

Next, the constants used to determine the antenna pattern factor are computed. First, if a user-defined height-finder antenna pattern has been specified, ($i_{pat} = 6$), along with power cut-back angles and factors, then the angles are converted to radians and stored in the array, *hfangr*. If the cut-back angles are not steadily increasing, i_{error} is set to -7 and the CSC is exited; otherwise, the CSC proceeds with the next step.

If a directional antenna pattern has been specified, the antenna vertical beamwidth in degrees, μ_{bw}, is checked for extremely small beamwidth values. If the value is less than or equal to 10^{-4}, i_{error} is set to -10 and the CSC is exited; otherwise, the CSC proceeds with the next step.

The antenna beamwidth and elevation angles are converted to radians (μ_{bwr} and μ_{or}, respectively) and the following variables, ant_{fac} and μ_{max}, for use in the ANTPAT SU are determined as follows. If the antenna pattern is Gaussian (i_{pat}=2), then ant_{fac} is given by

$$ant_{fac} = \frac{.34657359}{\left[\mathrm{SIN}\left(\frac{\mu_{bwr}}{2}\right)\right]^2}.$$

If the antenna pattern is Sin(X)/X (i_{pat} = 3), or a generic height finder, (i_{pat} = 5), then ant_{fac} is given by

$$ant_{fac} = \frac{1.39157}{\mathrm{SIN}\left(\frac{\mu_{bwr}}{2}\right)},$$

and μ_{max} is given by

$$\mu_{max} = \mathrm{TAN}^{-1}\left(\frac{\pi}{ant_{fac}\sqrt{1 - \left(\frac{\pi}{ant_{fac}}\right)^2}}\right).$$

Next, the TERINIT SU is referenced to initialize all terrain profile and associated arrays. If an error has occurred while in the TERINIT SU, i_{error} is returned with a non-zero value and the CSC is exited, otherwise; the CSC proceeds with the next step.

The variable y_{fref} is initialized to 0. If a terrain profile has been specified, (f_{ter}='.true.'), then y_{fref} is set equal to ty_1. Next, the following height arrays are initialized as follows:

$$z = hm_{ref} + i\Delta z_{out};$$

$$zout_i = z,$$

$$zro_i = z - y_{fref},$$

$$zoutma_i = z - ant_{ref},$$

$$zoutpa_i = z - y_{fref} + ant_{ht},$$

where the index, i, ranges from 0 to n_{zout}.

The GETMODE SU is then referenced to determine the execution mode the APM CSCI will operate. Next, the REFINIT SU is referenced to initialize all refractivity associated arrays. If an error has occurred while in the REFINIT SU, i_{error} is returned with a non-zero value and the CSC is exited; otherwise, the CSC proceeds to the next step. If the flag, i_{tropo}, equals 1, then the TROPOINIT SU is referenced to initialize all arrays associated with troposcatter calculations.

The limiting grazing angle, ψ_{lim}, is computed as

$$\psi_{lim} = \mathrm{AMAX}\left(.002, \frac{.04443}{f_{MHz}^{.3333}}\right).$$

If more than one refractivity profile has been specified (n_{prof}>1), then ψ_{lim} is multiplied by 2. It is then adjusted for trapping effects by

$$\psi_{lim} = \psi_{lim} + \sqrt{\left|2\left(r_{mmax} - r_{mmin}\right)\right|} \ ,$$

where r_{mmax} and r_{mmin} are determined in the REFINIT SU. The RO elevation angle limit, α_{lim}, is given by

$$\alpha_{lim} = \sqrt{\left|\psi_{lim}^2 + 2\left(rm_{i_{start}} - rm_0\right)\right|},$$

where the array, rm, and index, i_{start}, are determined in the REFINIT SU.

Next, several variables are initialized. The height tolerance, z_{tol}, is initialized to 0.05 and the minimum power of 2 transform, ln_{min}, is initialized to 10. If no terrain profile is specified and f_{MHz} is less than or equal to 3001 MHz, then ln_{min} is set equal to 9. The range and index variables for the RO region, i_{ROp} and x_{ROn}, are initialized to -1 and 0, respectively.

The minimum height for the PE calculation region is determined next. The minimum height encompassing all trapping refractive layers is given by

$$h_{test} = h_{trap} + h_{thick} \ ,$$

where h_{trap} and h_{thick} are determined in the REFINIT SU. If running in the full hybrid mode, (i_{hybrid}=1), the minimum height for the PE calculation region is given by

$$z_{test} = \mathrm{AMAX}\left(h_{test}, 1.2\, h_{termax}\right),$$

where h_{termax} is determined in the TERINIT SU. If running in either the partial hybrid mode or PE-only mode, z_{test} is then given by

$$z_{test} = \mathrm{AMAX}\left(ht_{lim}, ant_{ref}\right).$$

The tangent angle, a_{test}, used for automatic calculation of the maximum propagation angle, is also given by

$$a_{test} = \mathrm{TAN}^{-1}\left(\frac{z_{test} + ant_{ref} + \dfrac{r_{max}^2}{a_{ek2}}}{r_{max}}\right),$$

with α_{lim} now set equal to the greater of a_{test} or the previously determined α_{lim}.

The maximum propagation angle in the PE region, Θ_{max}, is now determined by referencing the GETTHMAX SU. If i_{hybrid} equals 2 (partial hybrid mode) and z_{lim} is greater than ht_{lim}, then i_{hybrid} is set equal to 1 (PE-only mode). Next, z_{lim} is set equal to the minimum of ht_{lim} and z_{lim}.

The following procedure is performed to maximize Θ_{max} within the given transform size as determined in the FFTPAR SU (referenced from the GETTHMAX SU). This procedure is performed only if a terrain profile is specified, if running in an execution mode other than the full hybrid mode, and if $0.74z_{max}$ is greater than z_{lim}. The ratio of initial launch angle to maximum propagation angle, Θ_{frac}, is determined as

$$\Theta_{frac} = \frac{a_{launch}}{\Theta_{max}},$$

where a_{launch} is determined in the GETTHMAX SU. Next, the sine of Θ_{max} is computed as

$$z_{max} = \frac{z_{lim}}{.74},$$

$$\mathrm{SIN}(\Theta_{max}) = \frac{n_{fft}\,\lambda}{2\,z_{max}}.$$

Upper limits on the sine of Θ_{max} are imposed according to

$$\mathrm{SIN}(\Theta_{max}) = \mathrm{AMIN}(\mathrm{SIN}(\Theta_{max}), \mathrm{SIN}(10°)); \quad \textit{for } f_{MHz} > 1000,$$

$$\mathrm{SIN}(\Theta_{max}) = \mathrm{AMIN}(\mathrm{SIN}(\Theta_{max}), \mathrm{SIN}(15°)); \quad \textit{for } f_{MHz} \le 1000.$$

Θ_{max} is then recomputed, along with other corresponding PE calculation parameters:

$$\Delta z_{PE} = \frac{\lambda}{2\mathrm{SIN}(\Theta_{max})},$$

$$\Theta_{max} = \mathrm{SIN}^{-1}(\Theta_{max}),$$

$$z_{max} = n_{fft}\,\Delta z_{PE},$$

$$\Theta_{75} = .75\,\Theta_{max},$$

with the launch angle recomputed as

$$a_{launch} = \Theta_{frac}\Theta_{max}.$$

Next, the index of the output range step, i_{xostp}, at which the XO model will be applied, is initialized to 0. The following steps (1 through 6) are performed if i_{hybrid} is not equal to 0.

1. If z_{lim} is less than ht_{lim}-10^{-3}, then the SU proceeds with steps 2 through 6. Otherwise, these steps are skipped and the output range and index, r_{atz} and i_{ratz}, respectively, are calculated as

$$r_{atz} = 2\,r_{max},$$

$$i_{ratz} = n_{rout} + 1.$$

2. The bin number, jz_{lim}, corresponding to z_{lim}, is given by

$$jz_{lim} = \mathrm{INT}\left(\frac{z_{lim}}{\Delta z_{PE}}\right),$$

and z_{lim} is recomputed such that it corresponds to an integer multiple of bins or mesh heights, $z_{lim} = j z_{lim} \Delta z_{PE}$.

3. Next, r_{atz} and i_{ratz} are determined based on the height, angle, and range arrays, *htemp*, *raya*, and *rtemp*, previously determined in the GETTHMAX SU. First, the index, *j*, is initialized to i_{ap} (previously determined in the GETTMAX SU) and the index, *id*, is initialized to 1. The following steps (a through b) are repeated until *j* is greater than i_{rtemp}.
   a. If $htemp_j$ is greater than z_{lim}, then the iteration is ended and the SU proceeds with step 4.
   b. If $htemp_j$ is greater than zrt_{id}, then *id* is incremented by 1. The index *j* is now incremented by 1. Steps 3a through 3b are then repeated.
4. The index, *ira*, is set equal to the greater of 1 or *j*-1; the index *idg* is set equal to *id*-1; and the gradient g_{rd} is set equal to gr_{idg}.
5. Next, the ray with initial launch angle, a_{launch}, is traced from height, $htemp_{ira}$, to z_{lim}. The square of the local ray angle, *rad*, at the end of the ray trace step is given by

$$rad = raya_{ira}^2 + 2\, g_{rd} \left(z_{lim} - htemp_{ira} \right).$$

   The local ray angle, a_{atz}, at height, z_{lim}, is initialized to 0. If *rad* is greater than 0, then a_{atz} is given by

$$a_{atz} = \mathrm{SIGN}(1, raya_{ira}) \sqrt{rad},$$

   and the range, r_{atz}, is now given by

$$r_{atz} = rtemp_{ira} + \frac{a_{atz} - raya_{ira}}{g_{rd}}.$$

6. If r_{atz} is less than r_{max} and z_{lim} is less than ht_{lim}, then the index *k* is determined such that $rngout_k > r_{atz}$ and $rngout_{k-1} < r_{atz}$. Then i_{ratz} is set equal to the smaller of n_{rout} and *k*, and i_{xostp} is set equal to i_{ratz}.

Next, the radar horizon range, r_{hor}, for the source height, ant_{ref}, and 0 receiver height, is computed by

$$r_{hor} = 4124.5387 \sqrt{ant_{ht}},$$

and the initial PE range step is given by

$$\Delta r_{PE} = 2\, k_o\, \Delta z_{PE}^2 .$$

Due to numerical constraints, numerical limits will be imposed on the PE range step, depending on r_{max} as follows. If performing a terrain case, Δr_{PE} is set equal to the smaller of Δr_{PE} and 700; and the variable rl_{lim} is given by

$$rl_{lim} = 75 \qquad \text{for } 5\ km \le r_{max} < 10\ km;$$

$$rl_{lim} = 90 \qquad \text{for } 10\ km \le r_{max} < 15\ km;$$

$$rl_{lim} = 100 \qquad \text{for } 15\ km \le r_{max} < 20\ km;$$

rl_{lim}=110 for $20\ km \le r_{max} < 30\ km$;

rl_{lim}=175 for $30\ km \le r_{max} < 50\ km$;

rl_{lim}=200 for $50\ km \le r_{max} < 75\ km$;

rl_{lim}=250 for $75\ km \le r_{max} < 100\ km$;

rl_{lim}=300 for $100\ km \le r_{max}$.

The variable, Δr_{PE}, is then set equal to the greater of Δr_{PE} and rl_{lim}. If r_{fix} (previously determined in the TERINIT SU) is greater than 0, then the temporary range step variable, r_d, is given by

$$r_d = \frac{r_{fix}}{\Delta r_{PE}},$$

and Δr_{PE} is recomputed according to

$$\Delta r_{PE} = \mathrm{NINT}\left(\frac{1}{r_d}\right) r_{fix}; \quad for \ \ r_d < 1,$$

$$\Delta r_{PE} = \frac{r_{fix}}{\mathrm{NINT}(r_d)}; \quad for \ r_d \ge 1.$$

The variable, iz_{inc}, is then initialized to 1.

If no terrain profile is specified, then Δr_{PE} is determined as follows. If running in the PE-only mode, Δr_{PE} is given by

$$\Delta r_{PE} = \mathrm{AMAX}(\mathrm{AMIN}(\Delta r_{PE}, 1000), 30).$$

Next, if r_{max} is greater or equal to r_{hor}, then Δr_{PE} is set equal to the greater of 300 and Δr_{PE}. The variable, iz_{inc}, is then initialized to 3. If $f_{MHz} \ge$ 10,000 MHz, then iz_{inc} is set equal to 1; if 5,000 $\le f_{MHz} <$ 10,000 MHz, then iz_{inc} is set equal to 2.

All variables and arrays associated with XO calculations are now initialized, provided i_{xostp} is greater than 0. The maximum number of points, iz_{max}, allocated for arrays used in XO calculations, is determined by

$$iz_{max} = \frac{\mathrm{NINT}\left(\frac{r_{max} - r_{atz}}{\Delta r_{PE}}\right)}{iz_{inc}} + 4.$$

Next, variables needed for spectral estimation calculations are initialized. The number of bins, n_p, considered in the upper PE region, is set equal to 8 if no terrain profile is specified, and 16, otherwise. The power of 2 transform, ln_p, is set equal to 6 if no terrain profile is specified, and 7, otherwise. The following variables are given by

$$n_s = 2^{ln_p},$$

$$n_{p4} = \frac{n_p}{4},$$

$$n_{p34} = 3n_{p4},$$

$$cn_{p75} = \frac{\pi}{n_{p4}}.$$

The ALLARRAY_XO SU is then referenced to allocate and initialize all arrays associated with XO calculations. The filter array, *filtp*, is now determined by

$$filtp_i = .5 + .5\mathrm{COS}(i\, cn_{p75}); \quad for\ i = 0,1,2,\ldots,n_{p4},$$

and the variable, xo_{con}, is given by

$$xo_{con} = \frac{\lambda}{2n_s \Delta z_{PE}}$$

One-half the PE range step, Δr_{PE2}, is given by ½ Δr_{PE}, and the following PE transform variables are computed: the angle (or p-space) mesh size, Δp; the Fourier transform normalization constant, f_{norm}; the constant used in determining the free-space phase factors, c_n, the transform size minus 1, n_{m1}; ¾ of the transform size, $n_{3/4}$; and twice the height (or z-space) mesh size, Δz_{PE2}, are determined as follows:

$$\Delta p = \frac{\pi}{z_{max}}, \quad f_{norm} = \frac{2}{n_{fft}},$$

$$c_n = \frac{\Delta p}{k_o}, \quad n_{m1} = n_{fft} - 1, \; n_4 = \frac{n_{fft}}{4},$$

$$n_{3/4} = 3n_4, \; \Delta z_{PE2} = 2\, \Delta z_{PE}.$$

The ALLARRAY_PE SU is then referenced to allocate and initialize all arrays associated with PE calculations. The filter array, *filt*, for subsequent filtering of the PE field, is given by

$$filt_i = .5 + .5\mathrm{COS}\left(i\frac{\pi}{n_4}\right), \quad for\ i = 0,1,2,\ldots,n_4.$$

The counter, i_g, indicating the current ground type being modeled, is initialized to 1. The DIEINIT SU is then referenced to initialize all dielectric ground constants. If f_{MHz} is less than or equal to 300 MHz and vertical polarization is specified, the GETALN SU is referenced to determine the surface impedance.

Next, the XYINIT SU is referenced to determine the initial PE solution, followed by a reference to the FFT SU to transform the PE field to z-space coordinates. If vertical polarization is specified, the variables C_1 and C_2 are initialized as follows. C_1 and C_2 are first given by

$$C_1 = .5\left(U_0 + U_{n_{fft}} root_{n_{fft}}\right) + \sum_{i=1}^{n_{m1}} U_i \, root_i ,$$

$$C_2 = .5\left(U_0 \, rootm_{n_{fft}} + U_{n_{fft}}\right) + \sum_{i=1}^{n_{m1}} U_{n_{fft}-i} \, rootm_i ,$$

where the arrays root and *rootm* are determined in the GETALN SU. The variables are further modified according to

$$C_1 = C_1 R, \quad C_2 = C_2 R,$$

where the coefficient, *R*, is also determined in the GETALN SU.

Next, several variables are initialized. The height of the ground, y_{last}, at the previous PE range is initialized to 0. If a terrain profile is specified, y_{last} is initialized to ty_1. The height of the ground, y_{curm}, midway between each PE range step, is initialized to 0, and the height of the ground, y_{cur}, at the current PE range, is initialized to 0. The PE mesh height array, *ht*, is given by

$$ht_i = i \, \Delta z_{PE}, \qquad for \;\; i = 0,1,2,\ldots,n_{fft} .$$

The index counter, *iz*, is initialized to 1 and the FILLHT SU is referenced to obtain the *htfe* array separating the FE from the RO region. Next, the free-space propagator array, *frsp*, is determined via a reference to the PHASE1 SU.

The following steps (**a** through **e**) are performed to adjust the refractivity arrays *gr*, *rm*, *q*, and *zrt*, associated with RO calculations for the special case when the terrain profile is initially flat, but at non-zero height.

a. First, the index *nlevel* is initialized to the number of refractivity levels, *levels*; y_{ref} is initialized to ty_1; *refref* and *href* are initialized to zero; and the index, *js*, is initialized to -1.

b. Next, *js* is determined such that $zrt_{js} < y_{ref} \leq zrt_{js+1}$. If a value for *js* is not found such that this condition holds true (i.e., *js* remains at -1), then the SU proceeds with step **e**.

c. The refractivity at y_{ref} is now computed from

$$f_{rac} = \frac{y_{ref} - zrt_{js}}{zrt_{js+1} - zrt_{js}} ,$$

$$r_{mu} = rm_{js} + \left(rm_{js+1} - rm_{js}\right) f_{rac} .$$

If INT(f_{rac}) is equal to 1, then *js* is set equal to *js*+1. The temporary counter, l_{new}, is initialized to *nlevel-js*.

d. The first element in *refref* and *href* is now set equal to r_{mu} and 0, respectively. The remainder of the current refractivity profile is adjusted in height and stored in *refref* and *href* according to

$$refref_j = rm_k$$
$$href_j = zrt_k - y_{ref} ; \quad for \; j = 1,2,3,\ldots l_{new} ,$$

where the index, k, is initialized to $js+1$ at the start and is incremented by one with each iteration of j. The variable, *levels*, indicating the number of levels in the newly created profile, is now set to l_{new}. *refref* and *href* are now used to initialize *rm* and *zrt*.

e. The arrays, *gr* and *q*, are now recomputed based on the newly adjusted refractivity arrays, *rm* and *zrt*. The gradient array, *gr*, is given by

$$gr_i = \frac{rm_{i+1} - rm_i}{zrt_{i+1} - zrt_i}; \quad for\, i = 0,1,2,\ldots, levels\,.$$

If the absolute value of gr_i is less than 10^{-8}, then gr_i is given by

$$gr_i = 10^{-8}\ \mathrm{SIGN}(1., gr_i)\,.$$

The array, q, is given by

$$q_i = 2(rm_{i+1} - rm_i); \quad for\, i = 0,1,2,\ldots, levels\,.$$

If a terrain profile is not specified and n_{prof} is equal to 1, then the INTPROF SU is referenced to determine the refractivity array *profint* at each PE mesh height. Next, the PHASE2 SU is referenced to determine the environmental phase array, *envpr*. Finally, if the absorption flag, k_{abs}, is equal to 1, then the GASABS SU is referenced to determine the absorption attenuation rate, gas_{att}. If k_{abs} is equal to 2, then gas_{att} is determined by the calling CSCI-specified attenuation rate, γ_a, multiplied by 10^{-2}.

Tables 8 and 9 identify, describe, and provide the units of measure and computational source for each input and output data element of the APMINIT CSC.

Table 8. APMINIT CSC input data element requirements.

Name	Description	Units	Source
abs_{hum}	Absolute humidity near the surface	g/meter^3	Calling CSCI
a_{ek}	$^4/_3$ effective earth's radius	meters	APM_MOD
ant_{ht}	Transmitting antenna height above local ground	meters	Calling CSCI
dielec	Two-dimensional array containing the relative permittivity and conductivity; $dielec_{1,i}$ and $dielec_{2,P}$, respectively.	N/A, S/m	Calling CSCI
f_{MHz}	Frequency	MHz	Calling CSCI
γ_a	Surface specific attenuation	dB/km	Calling CSCI
hfang	Cut-back angles	degrees	Calling CSCI
hffac	Cut-back antenna pattern factors	N/A	Calling CSCI
h_{max}	Maximum output height with respect to mean sea level	meters	Calling CSCI
h_{min}	Minimum output height with respect to mean sea level	meters	Calling CSCI

Table 8. APMINIT CSC input data element requirements. (Continued)

Name	Description	Units	Source
hmsl	Two-dimensional array containing heights with respect to mean sea level of each profile. Array format must be $hmsl_{i,j}$ = height of i^{th} level of j^{th} profile; $j = 1$ for range-independent cases	meters	Calling CSCI
i_{extra}	Extrapolation flag for refractivity profiles entered below mean sea level $i_{extra} = 0$; extrapolate to minimum terrain height standard atmosphere gradient $i_{extra} = 1$; extrapolate to minimum terrain height using first gradient in profile	N/A	Calling CSCI
i_{gr}	Number of different ground types specified	N/A	Calling CSCI
igrnd	Integer array containing ground type composition for given terrain profile—can vary with range. Different ground types are: 0 = Seawater 1 = Freshwater 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1 degree C 6 = Ice at -10 degree C 7 = User defined (in which case, values of relative permittivity and conductivity must be given).	N/A	Calling CSCI
i_{pat}	Antenna pattern type $i_{pat} = 1$: Omni-directional $i_{pat} = 2$: Gaussian $i_{pat} = 3$: Sine(x)/x $i_{pat} = 4$: Cosecant-squared $i_{pat} = 5$: Generic height-finder $i_{pat} = 6$: User-defined height-finder	N/A	Calling CSCI
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
i_{rtemp}	Temporary number of range steps (used for ray tracing)	N/A	APM_MOD
i_{tp}	Number of height/range points in profile	N/A	Calling CSCI

Table 8. APMINIT CSC input data element requirements. (Continued)

Name	Description	Units	Source
i_{tropo}	Troposcatter calculation flag: i_{tropo} = 0; no troposcatter calcs i_{tropo} = 1; troposcatter calcs	N/A	Calling CSCI
lvlp	Number of levels in refractivity profile	N/A	Calling CSCI
μ_{bw}	Antenna vertical beamwidth	degrees	Calling CSCI
μ_o	Antenna elevation angle	degrees	Calling CSCI
n_{facs}	Number of user-defined cut-back angles and cut-back antenna factors for user specified height-finder antenna type	N/A	Calling CSCI
n_{prof}	Number of refractivity profiles	N/A	Calling CSCI
n_{rout}	Number of output height points desired	N/A	Calling CSCI
n_{zout}	Number of output range points desired	N/A	Calling CSCI
pi	Constant equal to the value of π	N/A	APM_MOD
r_{adc}	Radians to degrees conversion factor	dgrees/ radians	APM_MOD
refmsl	Two-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be $refmsl_{i,j}$ = M-unit at i^{th} level of j^{th} profile; j = 1 for range-independent cases	M-units	Calling CSCI
rgrnd	Array containing ranges at which varying ground types apply.	meters	Calling CSCI
r_{max}	Maximum specified range	meters	Calling CSCI
rngprof	Ranges of each profile; $rngprof_i$ = range of i^{th} profile	meters	Calling CSCI
t_{air}	Air temperature near the surface	°C	Calling CSCI
terx	Range points of terrain profile	meters	Calling CSCI
tery	Height points of terrain profile	meters	Calling CSCI

Table 9. APMINIT CSC output data element requirements. (Continued)

Name	Description	Units
a_{atz}	Local ray or propagation angle at height, z_{lim}, and range r_{atz},	radians
a_{ek2}	Twice $^4/_3$ effective earth's radius	meters
afac	Antenna pattern parameter (depends on i_{pat} and μ_{bw})	N/A
a_{launch}	Launch angle used which, when traced, separates PE and XO regions from the RO region	N/A
α_{lim}	Elevation angle of the RO limiting ray	radians
C_1	Coefficient used in vertical polarization calculations	N/A
C_2	Coefficient used in vertical polarization calculations	N/A
c_n	Constant equals $\Delta p / k_o$	N/A
con	$10^{-6} k_o$	meters^{-1}
Δp	Mesh size in angle- (or p-) space	radians/ meters
Δr_{out}	Output range step	meters
Δr_{PE}	PE range step	meters
Δr_{PE2}	½ PE range step	meters
Δz_{PE}	PE mesh height increment (bin width in z-space)	meters
Δz_{PE2}	$2\ \Delta z_{PE}$	meters
Δz_{out}	Output height increment	meters
dielec	Two-dimensional array containing the relative permittivity and conductivity, $dielec_{1,i}$ and $dielec_{2,i}$, respectively.	N/A, S/m
filt	Cosine-tapered (Tukey) filter array	N/A
filtp	Array filter for spectral estimation calculations	N/A
f_{norm}	Normalization factor	N/A
fslr	Free space loss array for output ranges	dB
gas_{att}	Gaseous absorption attenuation rate	DB/km
gr	Intermediate M-unit gradient array, RO region	M-units/ meter
hfangr	Cut-back angles	Radians
ht	PE mesh height array of size n_{fft}	meters
i_{error}	Error flag	N/A

Table 9. APMINIT CSC output data element requirements. (Continued)

Name	Description	Units
i_g	Counter indicating current ground type being modeled	N/A
i_{ratz}	Index of output range step in which to begin storing propagation factor and outgoing angle for XO region	N/A
$iROp$	Array index for previous range in RO region	N/A
i_{tpa}	Number of height/range points pairs in profile tx, ty	N/A
i_{xo}	Number of range steps in XO calculation region	N/A
i_{xostp}	Current output range step index for XO calculations	N/A
iz	Number of propagation factor, range, and angle triplets stored in $ffacz$	N/A
iz_{inc}	Integer increment for storing points at top of PE region (i.e., points are stored at every iz_{inc} range step)	N/A
jz_{lim}	PE bin # corresponding to z_{lim}, (i.e., $z_{lim} = jz_{lim}\,\Delta z_{PE}$)	N/A
k_{abs}	Gaseous absorption calculation flag: k_{abs} = 0; no absorption loss k_{abs} = 1; compute absorption loss based on air temperature, t_{air}, and absolute humidity, abs_{hum} k_{abs} = 2; compute absorption loss based on specified absorption attenuation rate, γ_a	N/A
k_o	Free space wavenumber	meters^{-1}
$levels$	Number of levels in gr, q, and zrt arrays	N/A
λ	Wavelength	meters
ln_{min}	Minimum power of 2 transform size	N/A
ln_p	Power of 2 transform size used in spectral estimation calculations (i.e., $n_p = 2^{lnp}$)	N/A
μ_{or}	Antenna pattern elevation angle	radians
μ_{bwr}	Antenna vertical beamwidth in radians	radians
μ_{max}	Limiting angle for SIN(X)/X and generic height finder antenna pattern factors	N/A
$n_{3/4}$	$\frac{3}{4}\, n_{fft}$	N/A
n_4	$\frac{1}{4}\, n_{fft}$	N/A
n_{m1}	$n_{fft} - 1$	N/A
n_p	Number of bins in upper PE region to consider for spectral estimation.	N/A

Table 9. APMINIT CSC output data element requirements. (Continued)

Name	Description	Units
n_{p34}	¾ n_p	N/A
n_{p4}	¼ n_p	N/A
n_s	Transform size for spectral estimation calculations	N/A
p_{elev}	Sine of antenna elevation angle	N/A
pl_{cnst}	Constant used in determining propagation loss (pl_{cnst} = 20 LOG(2 k_o))	N/A
ψ_{lim}	Grazing angle of limiting ray	radians
q	Intermediate M-unit difference array, RO region	M-units
r_{atz}	Range at which z_{lim} is reached (used for hybrid model)	meters
rlogo	Array containing 20 times the logarithm of all output ranges	N/A
rm	Intermediate M-unit array, RO region	M-units
rngout	Array containing all desired output ranges	meters
rsqrd	Array containing the square of all desired output ranges	meters^2
s_{bw}	Sine of antenna vertical beam width	N/A
Θ_{max}	Maximum propagation angle in PE calculations	radians
Θ_{75}	75% of maximum propagation angle in PE calculations	radians
xo_{con}	Constant used in determining ϑ_{out}	N/A
y_{cur}	Height of ground at current range r	meters
y_{curm}	Height of ground midway between last and current PE range	meters
y_{fref}	Ground elevation height at source	meters
y_{last}	Height of ground at previous range, r_{last}	meters
xROn	Next range in RO region	meters
z_{lim}	Height limit for PE calculation region	meters
z_{max}	Total height of the FFT/PE calculation domain	meters
zout	Array containing all desired output heights referenced to h_{minter}	meters
zoutma	Array output heights relative to "real" ant_{ref}	meters
zoutpa	Array output heights relative to "image" ant_{ref}	meters
zRO	Array of output heights in RO region	meters

Table 9. APMINIT CSC output data element requirements. (Continued)

Name	Description	Units
zrt	Intermediate height array, RO region	meters
z_{test}	Height in PE region that must be reached for hybrid model	meters
z_{tol}	Height tolerance for Newton's method	meters

5.1.1 Allocate Arrays APM (ALLARRAY_APM) SU

The ALLARRAY_APM SU allocates and initializes all dynamically dimensioned arrays associated with APM terrain, refractivity, troposcatter, and general variable arrays.

The ALLARRAY_APM SU utilizes the FORTRAN ALLOCATE and DEALLOCATE statements to dynamically size arrays previously declared with the ALLOCATABLE attribute in the APM_MOD MODULE or to free the array storage space previously reserved in an ALLOCATE statement. Each dimension of the ALLOCATABLE array is indicated by a colon in the APM_MOD module (e.g., *slp*(:)). The ALLOCATE statement establishes the upper and lower bounds of each dimension and reserves sufficient memory. Because attempting to allocate a previously allocated array causes a run-time error, each ALLOCATE statement for an array is preceded by a test to determine if it has been allocated. If it has, it is deallocated first before it is allocated.

Initially, the integer used to indicate an error, i_{error}, is set to zero. If in attempting to allocate an array, a value of i_{error} other than zero is returned by an ALLOCATE statement, then the SU is exited.

Note that each array that is dynamically allocated in this SU is initialized to zero.

Six integers input to this SU are used to dynamically allocate the arrays. Unless otherwise indicated, these integers are used as the single dimension of the dynamically allocated array. The first, i_{gr}, is the number of different ground types specified. The second, i_{tpa}, is the number of terrain points used internally in arrays *tx* and *ty*. The third, *lvlp*, is the number of levels in the refractivity profile. The fourth, n_{facs}, is the number of user-defined cut-back antenna pattern factors for the user-defined height-finder antenna type. The fifth, n_{rout}, is the integer number of output range points desired. And finally, the sixth, n_{zout}, is the integer number of output height points desired.

Table 10 provides the definitions of arrays allocated in this SU. The only array that is allocated using the integer n_{facs} is *hfangr*. The arrays that are allocated using the integer n_{rout} are: *rsqrd*, *fslr*, *rlogo*, and *rngout*. The arrays that are allocated using the integer n_{zout} are *zout*, *zro*, *zoutma*, *zoutpa*, *hlim*, *htfe*, *rfac*1, *rfac*2, and *rloss*.

The arrays associated with terrain information use either the integer i_{tpa} or the integer i_{gr}. The arrays that are allocated with the integer i_{tpa} are: *tx*, *ty*, and *slp*. The arrays allocated using the integer i_{gr} are *igrnd*, *rgrnd*, and *cn*2. The array *dielec* is allocated using two as the first dimension and i_{gr} as the second dimension.

The arrays associated with refractivity information use either the integer, *lvlp*, or the integer, *lvlpt*. The integer *lvlpt* is equal to the integer *lvlp* plus one. The arrays allocated using the integer *lvlp* are *refdum*, *htdum*, *href*, and *refref*. The arrays allocated using the integer *lvlpt* are *gr*, *q*, *rm*, and *zrt*.

The arrays associated with the troposcatter calculations use either integers i_{tpa}, n_{zout}, or n_{rout}. The arrays allocated using the integer i_{tpa} are *ad*1 and $\vartheta 1t$. The arrays allocated using the integer n_{zout} include *adif*, *d2s*, *rdt*, and $\vartheta 2s$.

Tables 10 and 11 identify, describe, and provide units of measure and computational source for each input and output data element of the ALLARRAY_APM SU.

Table 10. ALLARRAY_APM SU input data element requirements.

Name	Description	Units	Source
i_{gr}	Number of different ground types specified	N/A	Calling CSCI
i_{tpa}	Number of terrain points used internally in arrays tx and ty	N/A	APMINIT CSC
i_{tropo}	Troposcatter calculation flag: i_{tropo} = 0; no troposcatter calcs i_{tropo} = 1; troposcatter calcs	N/A	Calling CSCI
lvlp	Number of levels in refractivity profile	N/A	Calling CSCI
n_{facs}	Number of user-defined cut-back angles and cut-back antenna factors for user specified height-finder antenna type	N/A	Calling CSCI
n_{rout}	Number of output height points desired	N/A	Calling CSCI
n_{zout}	Number of output range points desired	N/A	Calling CSCI

Table 11. ALLARRAY_APM SU output data element requirements.

Name	Description	Units
ad1	Array of tangent ranges from source height—used with terrain profile	Meters
adif	Height differences between ant_{ref} and all output receiver heights	meters
nc^2	Array of complex dielectric constants	N/A
d2s	Array of tangent ranges for all output receiver heights over smooth surface	meters
dielec	Two-dimensional array containing the relative permittivity and conductivity, $dielec_{1,i}$ and $dielec_{2,i}$, respectively.	N/A, S/m
fslr	Free space loss array for output ranges	dB
gr	Intermediate M-unit gradient array, RO region	M-units/ meter
hfangr	Cut-back angles	radians

Table 11. ALLARRAY_APM SU output data element requirements. (Continued)

Name	Description	Units
hlim	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	meters
href	Heights of refractivity profile with respect to y_{ref}	meters
htdum	Height array for current interpolated profile	meters
htfe	Array containing the height at each output range separating the FE region from the RO region (full hybrid mode), or the FE region from the PE region (partial hybrid mode)	meters
i_{error}	Integer variable indicating error number for ALLOCATE and DEALLOCATE statements	N/A
igrnd	Integer array containing ground type composition for given terrain profile—can vary with range. Different ground types are: 0 = Seawater 1 = Freshwater 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1 degree C 6 = Ice at -10 degree C 7 = User defined (in which case, values of relative permittivity and conductivity must be given).	N/A
q	Intermediate M-unit difference array, RO region	M-units
rdt	Array of minimum ranges at which diffraction field solutions are applicable (for smooth surface) for all output receiver heights.	meters
refdum	M-unit array for current interpolated profile	M-units
refref	Refractivity profile with respect to y_{ref}	M-units
rfac1	Propagation factor at valid output height points from PE field at range, r_{last}.	dB
rfac2	Propagation factor at valid output height points from PE field at range, r	dB
rgrnd	Array containing ranges at which varying ground types apply.	meters
rlogo	Array containing 20 times the logarithm of all output ranges	N/A
rloss	Propagation loss	dB
rm	Intermediate M-unit array, RO region	M-units

Table 11. ALLARRAY_APM SU output data element requirements. (Continued)

Name	Description	Units
rngout	Array containing all desired output ranges	meters
rsqrd	Array containing the square of all desired output ranges	$meters^2$
slp	Slope of each segment of terrain	N/A
$\vartheta 1t$	Array of tangent angles from source height—used with terrain profile	radians
$\vartheta 0$	Array of angles used to determine common volume scattering angle	radians
$\vartheta 2s$	Array of tangent angles from all output receiver heights—used with smooth surface	radians
tx	Range points of terrain profile	meters
ty	Adjusted height points of terrain profile	meters
zout	Array containing all desired output heights referenced to h_{minter}	meters
zoutma	Array output heights relative to "real" ant_{ref}	meters
zoutpa	Array output heights relative to "image" ant_{ref}	meters
zRO	Array of output heights, RO region	meters
zrt	Intermediate height array, RO region	meters

5.1.2 Allocate Array PE (ALLARRAY_PE) SU

The ALLARRAY_PE SU allocates and initializes all dynamically dimensioned arrays associated with PE calculations.

The ALLARRAY_PE SU utilizes the FORTRAN ALLOCATE and DEALLOCATE statements to dynamically size arrays previously declared with the ALLOCATABLE attribute in the APM_MOD module or to free the array storage space previously reserved in an ALLOCATE statement. Each dimension of the ALLOCATABLE array is indicated by a colon in the APM_MOD MODULE (e.g., *slp*(:)). The ALLOCATE statement establishes the upper and lower bounds of each dimension and reserves sufficient memory. Because attempting to allocate a previously allocated array causes a run-time error, each ALLOCATE statement for an array is preceded by a test to determine if it has been allocated. If it has, it is deallocated first before it is allocated.

Initially, the integer used to indicate an error, i_{error}, is set to zero. If in attempting to allocate an array, a value of i_{error} other than zero is returned by an ALLOCATE statement, then the SU is exited.

Note that each array that is dynamically allocated in this SU is initialized to zero.

There are two integers input to this SU that are used to dynamically allocate the arrays. Unless otherwise indicated, these integers are used as the single dimension of the dynamically allocated array. The first, n_{fft}, is the transform size. The second, n_4, is the transform size n_4 divided by four.

Table 12 provides definitions of the arrays allocated in this SU. The arrays that are allocated using the integer, n_{fft}, are *root*, *rootm*, *envpr*, *frsp*, *U*, *Ulast*, *ht*, *profint*, *xdum*, *ydum*, *w*, and *ym*. The only array allocated using the integer n_4 is *filt*.

Tables 12 and 13 identify, describe, and show the units of measure and the computational source for each input and output data element respectively of the ALLARRAY_PE SU.

Table 12. ALLARRAY_PE SU input data element requirements.

Name	Description	Units	Source
n_{fft}	Transform size	N/A	FFTPAR SU
n_4	¼ n_{fft}	N/A	APMINIT SU

Table 13. ALLARRAY_PE SU output data element requirements.

Name	Description	Units
envpr	Complex [refractivity] phase term array interpolated every Δz_{PE} in height	N/A
filt	Cosine-tapered (Tukey) filter array	N/A
frsp	Complex free space propagator term array	N/A
ht	PE mesh height array of size n_{fft}	meters
i_{error}	Integer variable indicating error number for ALLOCATE and DEALLOCATE statements	N/A
profint	Profile interpolated to every Δz_{PE} in height	M-units
root	Array of R_T to the i^{th} power (e.g., $root_i = R_T^i$)	N/A
rootm	Array of $-R_T$ to the i^{th} power (e.g., $rootm_i = (-R_T)^i$)	N/A
U	Complex field at current PE range, r	µV/m
Ulast	Complex field at previous PE range, r_{last}	µV/m
w	Difference equation of complex PE field	µV/m
xdum	Real part of complex field array	µV/m
ydum	Imaginary part of complex field array	µV/m
ym	Particular solution of difference equation	N/A

5.1.3 Allocate Array XO (ALLARRAY_XO) SU

The ALLARRAY_XO SU allocates and initializes all dynamically dimensioned arrays associated with XO calculations.

The ALLARRAY_XO SU utilizes the FORTRAN ALLOCATE and DEALLOCATE statements to dynamically size arrays previously declared with the ALLOCATABLE attribute in the

APM_MOD MODULE or to free the array storage space previously reserved in an ALLOCATE statement. Each dimension of the ALLOCATABLE array is indicated by a colon in the APM_MOD module (e.g., *slp*(:)). The ALLOCATE statement establishes the upper and lower bounds of each dimension and reserves sufficient memory. Because attempting to allocate a previously allocated array causes a run-time error, each ALLOCATE statement for an array is preceded by a test to determine if it has been allocated. If it has, it is deallocated before it is allocated.

Initially, the integer used to indicate an error, i_{error}, is set to zero. If in attempting to allocate an array, a value of i_{error} other than zero is returned by an ALLOCATE statement, then the SU is exited.

Note that each array that is dynamically allocated in this SU is initialized to zero.

There are five integers input to this SU that are used to dynamically allocate the arrays. Unless otherwise indicated, these integers are used as the single dimension of the dynamically allocated array. The first of these is iz_{max}, the maximum number of points allocated for arrays associated with XO calculations. The second is n_{p4}, 1/4 of the number of points, n_p, used in the top part of the PE region for spectral estimation. The third is n_{rout}, the integer number of output range points desired. The fourth is *lvlp*, the number of points in the refractivity profile. And the last is n_s, the transform size used in spectral estimation calculations (i.e., $n_s=2^{ln_p}$). The integer, ln_p, is the power of 2 transform size used in spectral estimation calculations.

Table 14 provides definitions of the arrays allocated in this SU. The array, *ffrout*, is allocated using the integer two as the first dimension and the integer, n_{rout}, as the second dimension. The number three is used as the first dimension limit, and the integer, iz_{max}, is used as the second dimension in the allocation of the array, *ffacz*. Both of the arrays, *grad* and *htr*, are allocated with the first dimension given by *lvlp* and the second dimension given by iz_{max}. The array, *lvl*, is allocated using the integer, iz_{max}. The array, *filtp*, is allocated using the integer, n_{p4}. The three arrays, *xp*, *yp*, and *spectr* are allocated using the integer, n_s.

Tables 14 and 15 identify, describe, and provide the units of measure and computational source for each input and output data element of the ALLARRAY_XO SU.

Table 14. ALLARRAY_XO SU Input data element requirements.

Name	Description	Units	Source
iz_{max}	Maximum number of points allocated for arrays associated with XO calculations	N/A	APMINIT CSC
lvlp	Number of height/refractivity levels in profiles	N/A	Calling CSCI
n_{p4}	¼ n_p	N/A	APMINIT CSC
n_{rout}	Integer number of output range points desired	N/A	Calling CSCI
n_s	Transform size for spectral estimation calculations	N/A	APMINIT CSC

Table 15. ALLARRAY_XO SU Output data element requirements.

Name	Description	Units
ffrout	Array of propagation factors at each output range beyond r_{atz} and at height z_{lim}	dB
filtp	Array filter for spectral estimation calculations	N/A
grad	Two-dimensional array containing gradients of each profile used in XO calculations	M-units/ meter
htr	Two-dimensional array containing heights of each profile used in XO calculations	meters
i_{error}	Integer variable indicating error number for ALLOCATE and DEALLOCATE statements	N/A
lvl	Number of height levels in each profile used in XO calculations	N/A
spectr	Spectral amplitude of field	dB
xp	Real part of spectral portion of PE field	dB
yp	Imaginary part of spectral portion field	dB

5.1.4 Antenna Pattern (Antpat) SU

The ANTPAT SU calculates an antenna pattern factor (normalized antenna gain), $f(\alpha)$, for a specified antenna elevation angle, α. Currently, antenna pattern factors are included for six types of antennas. These patterns include an omni-directional (i_{pat}=1) type, a Gaussian (i_{pat}=2) type, a Sin(X)/X (i_{pat} =3) type, a cosecant-squared (i_{pat} =4) type, a generic height-finder (i_{pat} =5) type, and a user-defined height-finder type (i_{pat}= 6).

From two antenna pattern parameters, a_{fac} and p_{elev}, the antenna beam width, μ_{bwr}, and elevation angle, μ_{or}, a specified angle, α, for which the antenna pattern factor is desired, and the antenna radiation pattern type, i_{pat}, the antenna factor is calculated as follows.

If the antenna pattern is omni-directional, then $f(\alpha) = 1$. If the antenna pattern is Gaussian, then

$$f(\alpha) = e^{-a_{fac}(\mathrm{SIN}(\alpha)-p_{elev})^2}.$$

If the antenna pattern is cosecant-squared, compute the elevation angle relative to the antenna elevation angle as

$$\alpha_{pat} = \alpha - \mu_{or}.$$

The antenna pattern is now given as

$$f(\alpha) = \frac{s_{bw}}{\mathrm{SIN}(\alpha_{pat})} \qquad \text{for } \alpha_{pat} > \mu_{bwr},$$

$$f(\alpha) = \text{AMIN}\left(1,\ \text{AMAX}\left\{0.03, \left[1 + \frac{\alpha_{pat}}{\mu_{bwr}}\right]\right\}\right) \qquad \text{for } \alpha_{pat} < 0,$$

$$f(\alpha) = 1 \qquad \text{otherwise,}$$

where s_{bw} is determined in the APMINIT CSC.

If the antenna pattern is Sin(X)/X, a generic height-finder, or a user-specified height-finder, the following calculations are made.

1. The elevation angle relative to the antenna elevation angle, α_{pat}, is determined as in the previous definition. If the antenna radiation pattern type is a generic or user-specified height-finder, the radiation pattern is simulated as a Sin(X)/X type with the elevation angle adjusted to account for the current pointing angle of the main beam, χ. χ is set to the direct-path ray angle, α_d, if α_d is greater than the antenna elevation angle μ_{or}; otherwise, χ is set to the elevation angle. In this case, $\alpha_{pat} = \alpha - \chi$
2. The antenna pattern is now given as

$$f(\alpha) = 1 \qquad \text{for } |\alpha_{pat}| \le 10^{-6};$$

$$f(\alpha) = \frac{\text{SIN}(a_{fac}\ \text{SIN}(\alpha_{pat}))}{a_{fac}\ \text{SIN}(\alpha_{pat})}; \text{ for } |\alpha_{pat}| < \mu_{max},$$

$$\text{otherwise, } f(\alpha) = 0.$$

3. For a user-defined height-finder, the pattern factor is further adjusted by a power reduction factor, $hffac_i$, as

$$f(\alpha) = f(\alpha)\ hffac_i$$

where i is an angle counter, decremented by one from the number of power reduction angles, n_{facs}, for each power reduction angle, $hfangr_i$, which exceeds χ.

Tables 16 and 17 identify, describe, and provide the units of measure and computational source for each input and output data element of the ANTPAT SU.

Table 16. ANTPAT SU input data element requirements.

Name	Description	Units	Source
a_{fac}	Antenna pattern parameter (depends on i_{pat} and μ_{bw})	N/A	APMINIT CSC
α	Elevation angle at transmitter	radians	Calling SU
α_d	Direct ray elevation angle	radians	FEM SU ROCALC SU
$hfangr$	Cut-back angles	radians	Calling CSCI

Table 16. ANTPAT SU input data element requirements. (Continued)

Name	Description	Units	Source
$hffac$	Cut-back antenna pattern factors	N/A	Calling CSCI
i_{pat}	Antenna pattern type i_{pat} = 1: Omni-directional i_{pat} = 2: Gaussian i_{pat} = 3: Sine(x)/x i_{pat} = 4: Cosecant-squared i_{pat} = 5: Generic height-finder i_{pat} = 6: User-defined height-finder	N/A	Calling CSCI
μ_{or}	Antenna pattern elevation angle	radians	APMINIT CSC
μ_{bwr}	Antenna vertical beam width	radians	APMINIT CSC
μ_{max}	Limiting angle for Sin(X)/X and generic height finder antenna pattern factors	radians	APMINIT CSC
n_{facs}	Number of user-defined cut-back angles and cut-back pattern factors	N/A	Calling CSCI
p_{elev}	Sine of antenna elevation angle	N/A	APMINIT CSC
s_{bw}	Sine of antenna vertical beam width	N/A	APMINIT CSC

Table 17. ANTPAT SU output data element requirements.

Name	Description	Units
$f(\alpha)$	Antenna pattern factor for specified elevation angle, α	N/A

5.1.5 Dielectric Initialization (DieInit) SU

The DIEINIT SU determines the conductivity and relative permittivity as a function of frequency in MHz based on general ground composition types.

The DIEINIT SU supports the following general ground types: saltwater, freshwater, wet ground, medium dry ground, very dry ground, ice at -1°C, ice at -10°C, and user-defined. For all ground types other than "user-defined," the permittivity and conductivity are calculated as functions of frequency from curve fits to the permittivity and conductivity graphs shown in the Recommendations and Reports of the International Radio Consultative Committee (CCIR, 1986). For the i^{th} input ground type case, $igrnd_i$, the permittivity, ε_r, and conductivity, σ, are determined as described in the following subsections.

For salt water (case 0), the relative permittivity is given by 70 for $f_{MHz} \leq 2253.5895$, and the conductivity is given by 5.0 S/m for $f_{MHz} \leq 1106.207$. For $f_{MHz} > 2253.5895$, the relative permittivity is given by

$$\varepsilon_r = \begin{bmatrix} 1.4114535 \times 10^{-2} - 5.2122497 \times 10^{-8} f_{MHz} + 5.8547829 \times 10^{-11} f_{MHz}^2 \\ - 7.6717423 \times 10^{-16} f_{MHz}^3 + 2.9856318 \times 10^{-21} f_{MHz}^4 \end{bmatrix}^{-1}.$$

For $f_{MHz} > 1106.207$, the conductivity, σ, in S/m is given by

$$\sigma = \frac{3.8586749 + 9.1253873 \times 10^{-4} f_{MHz} + 1.530992 \times 10^{-8} f_{MHz}^2}{1.-2.1179295 \times 10^{-5} f_{MHz} + 6.5727504 \times 10^{-10} f_{MHz}^2 - 1.9647664 \times 10^{-15} f_{MHz}^3}.$$

For freshwater (case 1), the relative permittivity, ε_r, is given by 80 for $f_{MHz} \leq 6165.776$. For higher frequencies, ε_r is given by

$$\varepsilon_r = \frac{79.027635 - 3.5486605 \times 10^{-4} f_{MHz} + 8.210184 \times 10^{-9} f_{MHz}^2}{1.-2.2083308 \times 10^{-5} f_{MHz} + 2.7067836 \times 10^{-9} f_{MHz}^2 - 1.0007669 \times 10^{-14} f_{MHz}^3}.$$

For $f_{MHZ} > 5776.157$, the conductivity, σ, in S/m is given by

$$\sigma = \left(\frac{-.65750351 + 6.6113198 \times 10^{-4} f_{MHz} + 1.4876952 \times 10^{-9} f_{MHz}^2}{1.+5.5620223 \times 10^{-5} f_{MHz} + 3.0140816 \times 10^{-10} f_{MHz}^2} \right)^2.$$

For $f_{MHZ} \leq 5776.157$, the conductivity, σ, in S/m is given by

$$\sigma = \left(\frac{201.97103 + 1.2197967 \times 10^{-2} f_{MHz} - 1.728776 \times 10^{-6} f_{MHz}^2}{1.-2.5539582 \times 10^{-3} f_{MHz} - 3.7853169 \times 10^{5} f_{MHz}^2} \right)^{-1}.$$

For wet ground (case 2), the relative permittivity, ε_r, is given by 30 for $f_{MHz} \leq 1312.054$. For $1312.054 < f_{MHz} < 4228.11$, the relative permittivity, ε_r, is given by

$$\varepsilon_r = \sqrt{\frac{857.94335 + 5.5275278 \times 10^{-2} f_{MHz}}{1.-8.9983662 \times 10^{-5} f_{MHz} + 8.8247139 \times 10^{-8} f_{MHz}^2}}.$$

For $f_{MHz} \geq 4228.11$, the relative permittivity, ε_r, is given by

$$\varepsilon_r = \sqrt{\frac{915.31026 - 4.0348211 \times 10^{-3} f_{MHz} + 7.4342897 \times 10^{-7} f_{MHz}^2}{1. - 9.4530022 \times 10^{-6} f_{MHz} + 4.892281 \times 10^{-8} f_{MHz}^2}} .$$

For $f_{MHz} > 15454.4$, the conductivity, σ, in S/m for wet ground is given by

$$\begin{aligned}\sigma &= 0.8756665 + 4.7236085 \times 10^{-5} f_{MHz} + 2.6051966 \times 10^{-8} f_{MHz}^2 \\ &\quad - 9.235936 \times 10^{-13} f_{MHz}^3 + 1.4560078 \times 10^{-17} f_{MHz}^4 \\ &\quad - 1.1129348 \times 10^{-22} f_{MHz}^5 + 3.3253339 \times 10^{-28} f_{MHz}^6 .\end{aligned}$$

For $f_{MHz} \leq 15454.4$, the conductivity, σ, in S/m for wet ground is given by

$$\begin{aligned}\sigma &= 5.5990969 \times 10^{-3} + 8.7798277 \times 10^{-5} f_{MHz} + 6.2451017 \times 10^{-8} f_{MHz}^2 \\ &\quad - 7.1317207 \times 10^{-12} f_{MHz}^3 + 4.2515914 \times 10^{-16} f_{MHz}^4 \\ &\quad - 1.240806 \times 10^{-20} f_{MHz}^5 + 1.3854354 \times 10^{-25} f_{MHz}^6 .\end{aligned}$$

For medium dry ground (case 3), the relative permittivity, ε_r, is given by 15 for $f_{MHz} \leq 4841.945$. For $f_{MHz} > 4841.945$, the relative permittivity, ε_r, is given by

$$\varepsilon_r = \sqrt{\frac{215.87521 - 2.6151055 \times 10^{-3} f_{MHz} + 1.9484482 \times 10^{-7} f_{MHz}^2}{1. - 7.6649237 \times 10^{-5} f_{MHz} + 1.2565999 \times 10^{-8} f_{MHz}^2}} .$$

At $f_{MHz} \leq 4946.751$ for medium dry ground, the conductivity, σ, in S/m is given by

$$\begin{aligned}\sigma &= (2.4625032 \times 10^{-2} + 1.8254018 \times 10^{-4} f_{MHz} - 2.664754 \times 10^{-8} f_{MHz}^2 \\ &\quad + 7.6508732 \times 10^{-12} f_{MHz}^3 - 7.4193268 \times 10^{-16} f_{MHz}^4)^2 .\end{aligned}$$

For $f_{MHz} > 4946.751$, for medium dry ground, the conductivity σ in S/m is given by

$$\begin{aligned}\sigma &= (0.17381269 + 1.2655183 \times 10^{-4} f_{MHz} - 1.6790756 \times 10^{-9} f_{MHz}^2 \\ &\quad + 1.1037608 \times 10^{-14} f_{MHz}^3 - 2.9223433 \times 10^{-20} f_{MHz}^4)^2 .\end{aligned}$$

For very dry ground (case 4), the relative permittivity, ε_r, is given by 3 and the conductivity, σ, in S/m is 0.0001 for $f_{MHz} < 590.8924$. For $590.8924 \leq f_{MHz} \leq 7131.933$, the conductivity, σ, in S/m is given by

$$\sigma = 2.2953743 \times 10^{-4} - 8.1212741 \times 10^{-7} f_{MHz} + 1.8045461 \times 10^{-9} f_{MHz}^2$$
$$- 1.960677 \times 10^{-12} f_{MHz}^3 + 1.256959 \times 10^{-15} f_{MHz}^4 - 4.46811 \times 10^{-19} f_{MHz}^5$$
$$+ 9.4623158 \times 10^{-23} f_{MHz}^6 - 1.1787443 \times 10^{-26} f_{MHz}^7 + 7.9254217 \times 10^{-31} f_{MHz}^8$$
$$- 2.2088286 \times 10^{-35} f_{MHz}^9 .$$

For $f_{MHz} > 7131.933$ MHz, the conductivity σ in S/m is given by

$$\sigma = (-4.9560275 \times 10^{-2} + 2.9876572 \times 10^{-5} f_{MHz} - 3.0561848 \times 10^{-10} f_{MHz}^2 + 1.1131828 \times 10^{-15} f_{MHz}^3)^2 .$$

For ice at -1°C (case 5), the relative permittivity, ε_r, is 3 for all frequencies, and the conductivity, σ, for $f_{MHz} \leq 300$, is given by

$$\sigma = \frac{3.8814567 \times 10^{-5} + 9.878241 \times 10^{-6} f_{MHz} + 7.9902484 \times 10^{-8} f_{MHz}^2}{1 + 8.467523 \times 10^{-2} f_{MHz} - 9.736703 \times 10^{-5} f_{MHz}^2 + 3.269059 \times 10^{-7} f_{MHz}^3},$$

and for $f_{MHz} > 300$ is given by

$$\sigma = \frac{1.2434792 \times 10^{-4} + 8.680839 \times 10^{-7} f_{MHz} + 7.2701689 \times 10^{-11} f_{MHz}^2 - 2.6416983 \times 10^{-14} f_{MHz}^3 + 1.37552 \times 10^{-18} f_{MHz}^4}{1 + 2.824598 \times 10^{-4} f_{MHz} - 6.755389 \times 10^{-8} f_{MHz}^2 + 2.8728975 \times 10^{-12} f_{MHz}^3 - 1.8795958 \times 10^{-18} f_{MHz}^4}.$$

For ice at -10°C (case 6), the relative permittivity, ε_r, is 3 for all frequencies, and the conductivity, σ, for $f_{MHz} \leq 8753.398$, is given by

$$\sigma = \frac{1}{(51852.543 + 389.58894 f_{MHz})(1 - 8.1212741 \times 10^{-7} f_{MHz} + 6.832108 \times 10^{-5} f_{MHz}^2)},$$

and for $f_{MHz} > 8753.398$, is given by

$$\sigma = 4.13105 \times 10^{-5} + 2.03589 \times 10^{-7} f_{MHz} - 3.1739 \times 10^{-12} f_{MHz}^2 + 4.52331 \times 10^{-17} f_{MHz}^3 .$$

For the user-defined ground type (case 7), the relative permittivity, ε_r, and the conductivity, σ, in S/m are set equal to the input values, $dielec_{1,i}$ and $dielec_{2,i}$, respectively.

Finally, the complex dielectric constant is given by

$$nc_i^2 = \varepsilon_{ri} + 60\lambda\sigma_i; \quad for\ i = 1,2,3,\ldots i_{gr}.$$

Tables 18 and 19 identify, describe, and provide the units of measure and computational source for each input and output data element of the DIEINIT SU.

Table 18 DIEINIT SU input data element requirements.

Name	Description	Units	Source
dielec	Two-dimensional array containing the relative permittivity and conductivity, $dielec_{1,i}$ and $dielec_{2,i}$, respectively.	N/A, S/m	Calling CSCI, DIEINIT SU
f_{MHz}	Frequency	MHz	APM_MOD
i_{gr}	Number of different ground types specified	N/A	Calling CSCI
igrnd	Integer array containing ground type composition for given terrain profile – can vary with range. Different ground types are: 0 = Seawater 1 = Freshwater 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1 degree C 6 = Ice at -10 degree C 7 = User defined (in which case, values of relative permittivity and conductivity must be given).	N/A	Calling CSCI
rgrnd	Array containing ranges at which varying ground types apply.	meters	Calling CSCI

Table 19. DIEINIT SU output data element requirements.

Name	Description	Units
nc^2	Array of complex dielectric constants	N/A

5.1.6 Fast-Fourier-Transform (FFT) SU

The FFT SU separates the real and imaginary components of the complex PE field into two real arrays and then references the SINFFT SU to transform each portion of the PE solution.

For a transform size, n_{fft}, the real and imaginary parts of the complex PE field array, U, respectively, are found for the index, i, from 0 to n_{fft}:

$$xdum_i = \text{REAL}\left(U_i\right)$$

and

$$ydum_i = \text{IMAG}\left(U_i\right).$$

The SINFFT SU is referenced, in turn, for *xdum* and *ydum* along with ln_{fft}, the power of the transform size to the base 2 $\left(n_{fft} = 2^{ln_{fft}}\right)$. The real and imaginary parts of the resulting transform arrays are then converted to the complex array, *U*, for *i* equal 0 to n_{fft} by

$$U_i = \text{CMPLX}\left(xdum_i, ydum_i\right).$$

Tables 20 and 21 identify, describe, and provide units of measure and computational source for each input and output data element of the FFT SU.

Table 20. FFT SU input data element requirements.

Name	Description	Units	Source
ln_{fft}	Power of 2 transform size (i.e. $n_{fft} = 2^{ln_{fft}}$)	N/A	FFTPAR SU
n_{fft}	Transform size	N/A	FFTPAR SU
U	Complex field to be transformed	µV/m	Calling SU

Table 21. FFT SU output data element requirements.

Name	Description	Units
U	Transform of complex field	µV/m

5.1.7 FFT Parameters (FFTPAR) SU

The FFTPAR SU determines the required transform size based on the maximum PE propagation angle and the maximum height needed. If running in full or partial hybrid modes, the maximum height is the height necessary to encompass at least 20 percent above the maximum terrain peak along the path or the highest trapping layer specified in the environment profiles, whichever is greater. If running in a PE-only mode, the maximum height is the specified maximum output height.

For computational efficiency reasons, an artificial upper boundary is established for the PE solution. To prevent upward propagating energy from being "reflected" downward from this boundary and contaminating the PE solution, the PE solution field strength is attenuated or "filtered" above a certain height to ensure that the field strength just below this boundary is reduced to zero. The bin width in z-space, Δz_{PE}, is found from

$$\Delta z_{PE} = \frac{0.5\lambda}{\text{SIN}\left(\Theta_{\max}\right)},$$

where λ is the wavelength in meters and Θ_{max} is the maximum propagation angle in radians.

The flag, i_{flag}, is used to determine maximum FFT size based on a given Θ_{max} and the height needed to reach z_{lim}.

For $i_{flag} = 0$, the constants, ln_{fft}, n_{fft}, and z_{max} are found from ln_{min}:

$$ln_{fft} = ln_{min},$$
$$n_{fft} = 2^{ln_{fft}},$$
$$z_{max} = n_{fft}\Delta z_{PE},$$

where ln_{min} is the minimum power of 2 transform size. For smooth surface and frequencies less than or equal to 3000 MHz, ln_{min} is initialized to 9; for all other cases it is set to 10. Next, the transform size needed to perform calculations to a test height, z_t, is determined. First, z_t is set equal to z_{lim} minus 10^{-3}. Then a DO WHILE loop is executed as long as the condition ¾ $z_{max} < z_t$ is satisfied. Within this DO WHILE loop, z_{max} is found from

$$ln_{fft} = ln_{fft} + 1,$$
$$n_{fft} = 2^{ln_{fft}},$$
$$z_{maz} = n_{fft}\Delta z_{PE}.$$

For the case where i_{flag}=1, no iteration needs to be performed. The variable, z_{lim}, is determined by a given ln_{fft} and Θ_{max} from equations shown above.

Upon exiting, z_{lim} is computed as ¾ z_{maz}.

Tables 22 and 23 identify, describe, and provide units of measure and computational source for each input and output data element of the FFTPAR SU.

Table 22. FFTPAR SU input data element requirements.

Name	Description	Units	Source
i_{flag}	Flag indicating whether to determine maximum FFT size, n_{fft}, based on given Θ_{max} and z_{lim} or determine z_{lim} based on given Θ_{max} and FFT size n_{fft}.	N/A	Calling SU
λ	Wavelength	meters	Calling SU
ln_{min}	Minimum power of 2 transform size	N/A	Calling SU
Θ_{max}	Maximum propagation angle in PE calculations	radians	Calling SU
z_{lim}	Maximum height region where PE solution is valid	meters	Calling SU

Table 23. FFTPAR SU output data element requirements.

Name	Description	Units
Δz_{PE}	Bin width in z space	meters
ln_{fft}	Power of 2 transform size (i.e. $n_{fft} = 2^{ln_{fft}}$)	N/A
n_{fft}	Transform size	N/A
z_{lim}	Maximum height region where PE solution is valid	meters
z_{max}	Total height of the FFT/PE calculation domain	meters

5.1.8 Fill Height Arrays (FILLHT) SU

The FILLHT SU calculates the effective earth radius for an initial launch angle of 5° and fills an array with height values at each output range of the limiting submodel, depending on which mode is used. If running in a full hybrid mode, the array contains height values at each output range separating the PE from the RO region. If running in partial hybrid or PE-only modes, the array contains those height values at each output range at which the initial launch angle has been traced to the ground or surface. These height values represent the separating region where, above that height, valid loss is computed, and below that height, no loss is computed. This is done so that only loss values that fall within a valid calculation region are output.

Upon entering the SU, internal one-line ray trace functions are defined as

$$\mathrm{RADA1}(a,b) = a^2 + 2\,g_{rd}\,b,$$

$$\mathrm{RP}(a,b) = a + \frac{b}{g_{rd}},$$

$$\mathrm{AP}(a,b) = a + b\,g_{rd},$$

$$\mathrm{HP}(a,b) = a + \frac{b^2 - c^2}{2\,g_{rd}}$$

for general parameters a, b, c, and refractivity gradient, g_{rd}.

For the case when i_{hybrid}=1 (full hybrid mode), the height values at each output range separating the PE region from the RO region is determined. The ray defined by a 5° elevation angle is traced up to the maximum height, ht_{lim}, to define the effective earth radius. The initial angle and range, a_0 and r_0, at the start of the ray trace step are set equal to 5° and zero, respectively. The refractivity level index, i, is also initialized to zero. Then a DO WHILE loop is executed so long as the two conditions, ($zrt_{i+1} < ht_{lim}$) and ($i < levels$), are satisfied. Within this loop, the angle at the end of the trace, a_1, is found from

$$a_1 = \sqrt{\mathrm{RADA1}(a_0, zrt_{i+1} - zrt_i)}\ ,$$

where the gradient of the current refractivity layer being traced, g_{rd}, is given by gr_i. The range at the end of the trace, r_1, is found from

$$r_1 = \mathrm{RP}\left(r_0, a_1 - a_0\right).$$

At the end of the DO WHILE loop, a_0 is set equal to a_1, r_0 is set equal to r_1, and i is incremented by one. The DO WHILE loop is continually executed until one or both of the above conditions are no longer satisfied. Upon exit from the loop, g_{rd} is set to gr_i, and a_1 and r_1 are determined from the two expressions:

$$a_1 = \sqrt{\mathrm{RADA1}\left(a_0, ht_{lim} - zrt_i\right)},$$

$$r_1 = \mathrm{RP}\left(r_0, a_1 - a_0\right).$$

Then ta_{ek}, twice the effective earth's radius factor times the earth's radius, is computed for use in the FEM SU to correct heights for earth curvature and average refraction. It is given by

$$ta_{ek} = \frac{r_1^2}{ht_{lim} - a_5 r_1},$$

where a_5 is 5° expressed as radians (i.e, 0.087266 radians).

Finally, the height array, *htfe*, is determined as follows. The temporary variable, y_{ar}, is found from

$$y_{ar} = y_{fref} - ant_{ht},$$

where the parameter, y_{fref}, is the ground elevation height at the source, and ant_{ht} is the transmitting antenna height above the local ground at range 0. Then the values of $htfe_i$ are determined (with index, i, having values from 1 to n_{rout}) by

$$htfe_i = y_{fref}; \quad for\ rngout_i \le r_{tst}$$

$$htfe_i = \mathrm{AMIN}\left(ht_{lim}, \mathrm{AMAX}\left\{y_{fref}, y_{ar} + t_5\, rngout_i\right\}\right); \quad for\ rngout_i > r_{tst},$$

where r_{tst} is a constant range of 2,500 meters, t_5 is the tangent of 5°, and $rngout_i$ is the output range at every i^{th} range step.

For partial hybrid (PE plus XO) or PE-only modes, the initial launch angle is traced until it hits the surface, storing heights traced at each output range.

First, several variables are initialized. The angle at the start of the trace, a_0, is set to $-a_{launch}$ (determined in the GETTHMAX SU), the initial range, r_0, is set equal to zero, and the height at the start of the ray trace step, h_0, is set equal to ant_{ref}. The index, l, indicating the location of the source height in array, zrt, is set equal to the index, i_{start}. Finally, the index j is set equal to one.

The following steps (1 through 3) are performed until the ray has reached the surface or the ray has been traced to r_{max}, whichever comes first.

1. The output range to trace to r_o is initialized to $rngout_j$, and $htfe_j$ is initialized to 0.
2. Now, the following steps (a through c) are performed until r_0 has reached r_o, the ray has turned around within a refractive layer, or the ray has hit the surface, whichever comes first.

a. The range at the end of the ray trace step, r_1, is initialized to r_o. Then, if a_0 is less than zero, the refractivity gradient, g_{rd} is set equal to gr_{l-1}. The angle and height at the end of the trace step, a_1 and h_1, are now given by

$$a_1 = \mathrm{AP}(a_0, r_1 - r_0),$$
$$h_1 = \mathrm{HP}(h_0, a_1, a_0).$$

b. For negative angle values of a_1, the height, h_1, is now checked to determine if the ray has been traced through a lower refractive layer. If so, the index, l, is decremented by one and h_1 is now set to zrt_l. Finally, the variables, a_1 and r_1, are re-computed as

$$a_1 = -\sqrt{\mathrm{RADA1}(a_0, h_1 - h_0)},$$
$$r_1 = \mathrm{RP}(r_0, a_1 - a_0).$$

c. a_0, r_0, and h_0 are now set equal to the values of a_1, r_1, and h_1, respectively. If one of the conditions in step 2 has been met, then the SU proceeds to step 3; otherwise, steps 2a through 2c are repeated.

3. The ray has now been traced to the output range, $rngout_j$ and the height, h_0, at that range is stored in the $htfe_j$. The index, j, is incremented by one and if the ray has not reached the surface or r_{max}, then steps 1 through 3 are repeated.

Once the ray trace is completed, the index j is decremented by one and $htfe_j$ is set equal to hm_{ref} for all remaining output range steps j through n_{rout}.

Tables 24 and 25 identify, describe, and provide units of measure and computational source for each input and output data element of the FILLHT SU.

Table 24. FILLHT SU input data element requirements.

Name	Description	Units	Source
a_{launch}	Launch angle used which, when traced, separates the PE and XO regions from the RO region	radians	GETTHMAX SU
ant_{ht}	Transmitting antenna height above local ground	meters	Calling CSCI
ant_{ref}	Transmitting antenna height relative to the reference height, h_{minter}	meters	TERINIT SU
gr	Intermediate M-unit gradient array, RO region	(M-unit/m /meter)10^{-6}	REFINIT SU
hm_{ref}	Height relative to h_{minter}	meters	TERINIT SU
ht_{lim}	User-supplied maximum height relative to h_{minter}, i.e., $ht_{lim} = h_{max} - h_{minter}$	meters	TERINIT SU

Table 24. FILLHT SU input data element requirements. (Continued)

Name	Description	Units	Source
i_{hybrid}	Integer indicating which sub-models will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	N/A	GETMODE SU
i_{start}	Array index for height in RO region corresponding to ant_{ref}	N/A	REFINIT SU
$levels$	Maximum index of gr, q, and zrt arrays	N/A	REFINIT SU
n_{rout}	Integer number of the output range points desired	N/A	Calling CSCI
$rngout$	Array containing all output ranges	meters	APMINIT CSC
r_{tst}	Range set at 2.5 km to begin calculation of RO values	meters	APM_MOD
zrt	Height array used for RO calculations	meters	REFINIT SU
y_{fref}	Ground elevation height at the source	meters	APMINIT CSC

Table 25. FILLHT SU output data element requirements.

Name	Description	Units
$htfe$	Array of height values at each output range separating the PE region from the RO region	Meters
ta_{ek}	Twice the effective earth's radius	Meters

5.1.9 Gaseous Absorption (GASABS) SU

The GASABS SU computes the specific attenuation based on air temperature and absolute humidity. This SU is based on CCIR (International Telecommunication Union, International Radio Consultative Committee, now the ITU-R) Recommendation 676-1, "Attenuation by Atmospheric Gases in the Frequency Range 1-350 GHz."

The oxygen absorption for 15°C air temperature is computed from

$$\gamma_o = 10^{-3}\left(t_1 + t_2 + 0.00719\right)\left(\frac{f_{MHz}}{1000.}\right)^2,$$

where f_{MHz} is the frequency in MHz and the temporary variables, t_1 and t_2, are given by

$$t_1 = \frac{6.09}{\left(\frac{f_{MHz}}{1000.}\right)^2 + 0.227},$$

$$t_2 = \frac{4.81}{\left(\frac{f_{MHz}}{1000} - 57.0\right)^2} + 1.50.$$

A correction is made for the oxygen absorption for the actual air temperature, which is given by

$$\gamma_o = \left(1.0 + 0.01\left\{t_{air} - 15.0\right\}\right) \gamma_o ,$$

where t_{air} is the surface air temperature in degrees Centigrade.

The water vapor absorption is computed from

$$\gamma_w = \frac{\left(0.05 + 0.0021\, abs_{hum} + t_1 + t_2 + t_3\right)\left(\frac{f_{MHz}}{1000.}\right)^2 abs_{hum}}{10000.0},$$

where the temporary variables, t_1, t_2, and, t_3, are given respectively by

$$t_1 = \frac{3.6}{\left(\frac{f_{MHz}}{1000.} - 22.2\right)^2 + 8.5},$$

$$t_2 = \frac{10.6}{\left(\frac{f_{MHz}}{1000.} - 183.3\right)^2 + 9.0},$$

and

$$t_3 = \frac{8.9}{\left(\frac{f_{MHz}}{1000.} - 325.4\right)^2 + 26.3}.$$

The total specific absorption for sea-level air in dB/km multipied by a conversion factor for computing loss in dB is given by

$$gas_{att} = (\gamma_o + \gamma_w)10^{-2}.$$

Tables 26 and 27 identify, describe, and provide the units of measure and computational source for each input and output data element of the GASABS SU.

Table 26. GASABS SU input data element requirements.

Name	Description	Units	Source
abs_{hum}	Absolute humidity near the surface	g/meter^3	Calling CSCI
f_{MHz}	Frequency	MHz	Calling CSCI
t_{air}	Air temperature near the surface	°C	Calling CSCI

Table 27. GASABS SU output data requirements.

Name	Description	Units
gas_{att}	Gaseous absorption	dB/km

5.1.10 Get Alpha Impedance (GETALN) SU

The GETALN SU computes the surface impedance term in the Leontovich boundary condition and the complex index of refraction for finite conductivity. The implementation of these impedance formulas follow Kuttler and Dockery 's method (reference h).

Upon entering the SU, the complex refractive index, R_{ng}, is given by the square of the i_g^{th} complex refractive index:

$$R_{ng} = \sqrt{nc^2_{i_g}} ,$$

where nc^2 has been determined in the DIEINIT SU.

The surface impedance term, α_v, is given in terms of the complex refractive index, R_{ng}, and free-space wavenumber, k_o, for both vertical and horizontal polarization, by

$$\alpha_v = \frac{ik_o}{R_{ng}}; \quad for\, i_{pol} = 1,$$

$$\alpha_v = ik_o R_{ng}; \quad for\, i_{pol} = 0,$$

where i is the imaginary number $\sqrt{-1}$.

The determination of the complex root, R_T, of the quadratic equation for the mixed transform method is based on Kuttler's formulation. First, R_T is determined as follows. For horizontal polarization, R_T is given by

$$R_T = -\sqrt{1.0 + \left(\alpha_v \Delta z_{PE}\right)^2} - \alpha_v \Delta z_{PE} .$$

For vertical polarization, R_T is given by

$$R_T = \sqrt{1.0 + \left(\alpha_v \Delta z_{PE}\right)^2} - \alpha_v \Delta z_{PE} .$$

Next, the arrays, *root* and *rootm*, are determined by

$$root_i = R_T^i, \quad for\ i = 0,1,2,\ldots n_{fft}$$
$$rootm_i = \left(-R_T\right)^i, \quad for\ i = 0,1,2,\ldots n_{fft}\,.$$

The parameter, R, a coefficient used in the determination of C_1 and C_2 in the calling SU, is found from

$$R = \frac{2\left(1-R_T^2\right)}{\left(1+R_T^2\right)\left(1-R_T^{2n_{fft}}\right)}\,.$$

The parameters C_{1x} and C_{2x} are determined as a function of the range step, Δr_{PE}, from

$$C_{1x} = e^{\frac{i\Delta r_{PE}}{2k_0}\left(\frac{\mathrm{LN}(R_T)}{\Delta z_{PE}}\right)^2}$$

and

$$C_{2x} = e^{\frac{i\Delta r_{PE}}{2k_0}\left(\frac{\mathrm{LN}(R_T)-i\pi}{\Delta z_{PE}}\right)^2}\,.$$

Tables 28 and 29 identify, describe, and provide units of measure and computational source for each input and output data element of the GETALN SU.

Table 28. GETALN SU input data element requirements.

Name	Description	Units	Source
Δr_{PE}	PE range step	Meters	APMINIT CSC
Δz_{PE}	Bin width in z space	Meters	FFTPAR SU
i_g	Counter indicating current ground type being modeled	N/A	APMINIT CSC PESTEP SU
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
k_o	Free-space wavenumber	Meters^{-1}	APMINIT CSC
nc^2	Array of complex dielectric constants	N/A	DIEINIT SU
n_{fft}	Transform size	N/A	FFTPAR SU

Table 29. GETALN SU output data requirements.

Name	Description	Units
α_v	Surface impedance term	N/A
C_{1x}	Constant used to propagate C_1 by one range step	N/A
C_{2x}	Constant used to propagate C_2 by one range step	N/A
root	Array of R_T to the i^{th} power (e.g., $root_i = R_T^i$)	N/A
rootm	Array of $-R_T$ to the i^{th} power (e.g., $rootm_i = (-R_T)^i$)	N/A
R	Coefficient used in C_1 and C_2 calculations.	N/A
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	N/A

5.1.11 Get Mode (GETMODE) SU

The GETMODE SU determines what execution mode APM will run based on environmental inputs for the current application. Based on inputs, it determines whether to use the pure PE model (i_{hybrid}=0), full hybrid mode (i_{hybrid}=1), or partial hybrid mode (i_{hybrid}=2).

Initially, the variable, r_{flat}, indicating the maximum range at which the terrain profile remains flat from the source, is set equal to r_{max}. For antenna heights greater than 100 meters above the local ground height, i_{hybrid} is set equal to 0 and the SU is exited; otherwise, it must be determined whether to use full or partial hybrid modes.

For antenna heights less than 100 meters, i_{hybrid} is initialized to 1. If performing a smooth surface case (f_{ter}='.false.'), the SU is exited; otherwise, it proceeds with the next step. A test is made to see if the first 2500 meters of the terrain profile are flat. If it is not, then i_{hybrid} is set equal to 2 and the SU is exited. However, if the terrain profile is flat for *at least* the first 2500 meters, then an iteration is performed to determine the maximum range at which the profile remains flat. The variable, r_{flat}, is then set to this value, i_{hybrid} remains 1, and the SU is exited.

Tables 30 and 31 identify, describe, and provide units of measure and the computational source for each input and output data element of the GETMODE SU.

Table 30. GETMODE SU input data element requirements.

Name	Description	Units	Source
ant_{ht}	Transmitting antenna height above local ground	meters	Calling CSCI
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
i_{tpa}	Number of height/range points pairs in profile *tx, ty*	N/A	APMINIT CSC

Table 30. GETMODE SU input data element requirements. (Continued)

Name	Description	Units	Source
r_{max}	Maximum specified range	meters	Calling CSCI
r_{tst}	Range set at 2.5 km to begin calculation of RO values	meters	APM_MOD
slp	Slope of each segment of terrain	N/A	TERINIT SU
tx	Range points of terrain profile	meters	TERINIT SU

Table 31. GETMODE SU output data element requirements.

Name	Description	Units
i_{hybrid}	Integer indicating which sub-models will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	N/A
r_{flat}	Maximum range from the source at which the terrain profile remains flat	meters

5.1.12 Get Maximum Angle (GETTHMAX) SU

The GETTMAX SU performs an iterative ray trace to determine the minimum angle required (based on the reflected ray) in obtaining a PE solution. The determination of this angle depends on the particular mode of execution. For full and partial hybrid modes, a ray is traced up to a height that exceeds at least 20 percent above the maximum terrain peak along the path or the highest trapping layer specified in the environment profiles, whichever is greater. Heights and angles of this ray are stored at each output range. For the PE-only mode, a ray is traced for all heights up to the maximum output height. The maximum PE propagation angle, Θ_{max}, is then determined from the local maximum angle of the traced ray. For the full hybrid mode, the minimum PE propagation angle is required to meet the following criteria: (1) the top of the PE region must contain all trapping layers for all refractivity profiles; (2) the top of the PE region must be at least 20 percent higher than the highest peak along the terrain profile; and (3) the minimum PE propagation angle must be at least as large as the grazing angle of the limiting ray, ψ_{lim}.

First, four in-line ray-trace functions are defined for general parameters a, b, c, and g_{rd}:

$$\mathrm{RADA1}(a,b) = a^2 + 2\, g_{rd}\, b,$$

$$\mathrm{RP}(a,b) = a + \frac{b}{g_{rd}},$$

$$\mathrm{AP}(a,b) = a + b\, g_{rd}\,,$$

$$\mathrm{HP}(a,b,c) = a + \frac{b^2 - c^2}{2\, g_{rd}}.$$

The first parameter to be determined is the minimum PE angle limit, a_{mlim}. The parameter determined later, Θ_{max}, must be at least this value. The initial estimate of a_{mlim} is given by

$$a_{mlim} = \text{AMIN}\left(4, .37541 + 4.331e^{\frac{-f_{MHz}}{248.4}} + 1.42e^{\frac{-f_{MHz}}{2867}} + .4091e^{\frac{-f_{MHz}}{2495.}}\right),$$

$$a_{mlim} = r_{adc}\ a_{mlim}$$

where r_{adc} is the constant used to convert degrees to radians (i.e., .0174533). Next, a temporary variable, a_{ml}, is initialized to two times a_{mlim} if the polarization is vertical (i_{pol}=1), and 0, otherwise. If performing a terrain case (f_{ter} is '.true.') and using the full hybrid mode (i_{hybrid}=1), a_{ml} is further modified as a function of f_{MHz} according to

$$\begin{aligned} a_{ml} &= 2.5\,a_{mlim} && for\ f_{MHz} \le 1000, \\ a_{ml} &= 2\,a_{mlim} && for\ 1000 < f_{MHz} \le 1500, \\ a_{ml} &= 1.5\,a_{mlim} && for\ 1500 < f_{MHz} \le 2000. \end{aligned}$$

Finally, a_{mlim} is determined from

$$a_{mlim} = \text{AMAX}\left(a_{mlim}, a_{ml}\right).$$

Several constants needed in subsequent steps in this SU are determined. An initial estimate of the launch angle, a_{launch}, is initialized to α_{lim}, the elevation angle of the RO limiting ray. If using the full hybrid mode, then a_{launch} is set equal to the negative of a_{launch}. The maximum height to trace to z_{limt} is set equal to ht_{lim}-10^{-3}, and the range step, Δr_{temp}, for subsequent ray tracing is given by $r_{max}/200$. The terrain elevation height at the source, y_{nt}, is initialized to ty_1 provided APM is running in a full hybrid mode and ty_1 is greater than zero; otherwise, y_{nt} is initialized to 0.

An iterative ray trace determines the launch angle, a_{launch}, and subsequently Θ_{max} begins. The following steps (1 through 3) are performed until a ray has been safely traced from height ant_{ref} to z_{limt}.

1. At the start of the ray trace, the current local angle, (a_0); range, (r_0); height, (h_0); and refractive gradient index, (j) are initialized to a_{launch}, 0, ant_{ref}, and i_{start}, respectively. The counter index, k_t, for the terrain profile arrays, tx and ty, is initialized to one. The variable, r_o, the current output range to trace to, is set equal to zero. The following steps (a through d) are then performed for each ray trace step from 1 to i_{rtemp}.

   a. First, r_o is incremented by Δr_{temp}. Now steps (1) through (7) are performed until r_0 reaches r_o.

      (1) The range at the end of the ray trace step, r_1, is set equal to r_o, and the current refractive gradient, g_{rd}, is set equal to gr_j. If a_0 is less than zero, then gr_j is set equal to gr_{j-1}.

      (2) The angle at the end of the trace, a_1, is then given by

$$a_1 = \text{AP}\left(a_0, r_1 - r_0\right).$$

(3) If a_1 is of the opposite sign of a_0, then a_1 is set to zero, and r_1 is given by

$$r_1 = \mathrm{RP}(r_0, a_1 - a_0).$$

(4) The height at the end of the ray trace, h_1, is given by

$$h_1 = \mathrm{HP}(h_0, a_1, a_0).$$

(5) If a_1 is positive and h_1 has reached or surpassed the next height level, then a_1, r_1, j, and h_1, are found as follows. First, h_1 is set equal to zrt_j, and a_1 and r_1 are given by

$$\begin{aligned} a_1 &= \sqrt{\mathrm{RADA1}(a_0, h_1 - h_0)} \\ r_1 &= \mathrm{RP}(r_0, a_1 - a_0) \end{aligned},$$

then the index j is incremented by one, and the height, h_1, at the end of the ray trace step is given by

$$h_1 = \mathrm{AMIN}\left(ht_{lim}, zrt_j\right).$$

(6) However, if either of the conditions for a_1 and h_1 in step (5) are not met, and a_1 is less than or equal to 0, then h_1 is set equal to y_{nt} if the calculated value in step (4) is less than $y_{nt}+10^{-3}$. If the calculated value of h_1 in step (4) is less than $zrt_{j-1}+10^{-3}$, then h_1 is set equal to zrt_{j-1}, and j is set equal to $\mathrm{AMAX}(0, j-1)$. The variables, a_1 and r_1, are then determined from

$$\begin{aligned} a_1 &= -\sqrt{\mathrm{RADA1}(a_0, h_1 - h_0)}, \\ r_1 &= \mathrm{RP}(r_0, a_1 - a_0). \end{aligned}$$

(7) If the height at the end of the ray trace, h_1, is less than the height of the terrain, y_{nt} plus 10^{-3}, then the ray has hit the surface and is reflected. In this case, a_1 is set equal to minus a_1, ψ_{lim} is set equal to a_1, and the range, r_{pest}, at which loss values from the PE model will start being calculated, is set equal to r_1. In preparation for the next ray trace step, h_0 is set equal to h_1, r_0 is set equal to r_1, and a_0 is set equal to a_1. If the range r_1 is greater than r_{flat}, then the current iteration is exited and the SU proceeds to step b; otherwise, steps (1) through (7) are repeated until r_0 reaches r_o.

b. If running a terrain case (f_{ter} = '.true.'), at the end of the ray trace for the current step a check is made to see that the current height of the ray is at least 20 percent higher than the current terrain height. The counter, k_t, is determined such that $r_0 > tx_{k_t+1}$ and $k_t < i_{tpa}$. The height of the terrain, y_n, at the current range for the traced ray, is given by

$$y_n = 1.2\left(ty_{k_t} + slp_{k_t}\left(r_0 - tx_{k_t}\right)\right).$$

c. The ending angle, range, and height for each ray trace step is now stored in arrays *raya*, *rtemp*, and *htemp*, respectively.

d. Now, if running a full hybrid case ($i_{hybrid} = 1$), a test is made to determine if both h_0 is less than y_n and if r_0 is greater than r_{flat}. If these conditions are true, then the flag, i_{quit}, is set equal to 1. If the case is not a full hybrid case and if h_0 is less than y_n, then i_{quit} is set equal to 1. Finally, if h_0 is greater than or equal to z_{limt}, or i_{quit} equals 1, then the current iteration is exited and the SU proceeds to step 2; otherwise, steps a through d are repeated.

2. If the iteration defined by steps a through d has been prematurely terminated (i_{quit}=1), then the initial elevation angle, a_{launch}, is decreased by 10^3 radians for the full hybrid case (i_{hybrid}=1), and is increased by 10^{-3}, otherwise. If the previous iteration has not been prematurely terminated (i_{quit}=0), the SU continues with step 3.
3. If height z_{limt} is reached, then an initial launch angle (i.e., ray) has been found with all traced heights, ranges, and angles stored. The integer flag to continue ray tracing, i_{ray}, is set to equal 1 to terminate the iterative loop, and the index, i_{hmax}, indicating the range step at which z_{limt} is reached, is set equal to the range step index, i (the range step index counter in the iterative loop defined by steps 1 through 3).

The remaining elements from i_{hmax} to i_{rtemp} in arrays *htemp*, *rtemp*, and *raya* are filled with the values h_0, r_{max}, and a_0, respectively. Next, the index, i_{hmax}, is set equal to the minimum of i_{hmax} or i_{rtemp}.

The variable, Θ_{max}, is found for the PE region based on the local ray angles just determined for the particular ray traced. First, the index, i_{ap}, at which the local ray angle becomes positive (i.e., $raya_{iap}$) is determined. Next, several variables are initialized. The local indices, i_{ok} and i_{flag}, plus the variables, z_{lim} and a_{mxcur}, are each set equal to zero. The variable, a_{mxcur}, is the maximum local angle along the traced ray up to height, z_{lim}, with a minimum limit of a_{mlim}.

The variable, Θ_{max}, is then found from an iteration performed on the local angle and height at which the local maximum angle is reached. The following steps (1 through 6) are performed while the flag i_{ok} is 0.

1. The height in the PE region that must be reached for the hybrid model, z_t, is set equal to z_{test}-10^{-3}. Next, the first occurrence of $htemp_j$ that is greater than z_t is found and the index i_{st} is then set to the smaller of j or i_{hmax}.

2. The angle, a_{mxcur}, is now initialized to $|raya_1|$. The maximum angle in *raya* is then found looking only at elements from $raya_2$ to $raya_{ist}$ and a_{mxcur} is set equal to this angle.

3. a_{mxcur} is now set equal to the maximum of a_{mlim} and a_{mxcur}. The variable, a_{temp}, is now set to a_{mxcur} divided by 0.75. If running the PE-only mode (i_{hybrid}=2), z_{test} is given by

$$z_{test} = \mathrm{AMAX}\left(ant_{ref}, h_{test}, 1.2h_{termax}, 1000\right).$$

4. A reference is then made to the FFTPAR SU to determine new values for z_{test}, z_{max}, Δz_{PE}, ln_{fft}, and n_{fft} using the inputs: ln_{min}, λ, a_{temp}, and i_{flag}.

5. After the reference to the FFTPAR SU is made, if $i_{flag} = 0$, then it is set equal to 1. In addition, if not running a full hybrid case, i_{ok} is set equal to 1. However, if after the

reference to the FFTPAR SU is made, i_{flag} is equal to one and if the case is not a partial hybrid case; the iterative height tolerance tol is given by

$$tol = \frac{|z_{test} - z_{lim}|}{z_{test}} .$$

A test is then made to determine whether this value of tol is less than or equal to z_{tol}, the height tolerance for Newton's method. If it is, then the index, i_{ok}, is set equal to one.

6. Then z_{lim} is set equal to z_{test} and if i_{ok} is 0, steps 1 through 5 are repeated. Otherwise, the SU proceeds to the next step.

The variable, Θ_{75}, is now set equal to a_{mxcur}, and Θ_{max} is set equal to a_{temp}.

Before exiting this SU, the ray is traced again to each output range step, Δr_{out}, and heights are stored in the array, $hlim$. If running a full hybrid mode, the variables, a_0, r_0, h_0, and j are initialized to ψ_{lim}, r_{pest}, zero, and zero, respectively. If not running a full hybrid mode (i.e., $i_{hybrid} \neq 1$), then the variables, a_0, r_0, h_0, and j are set equal to a_{launch}, zero, ant_{ref}, and i_{start}, respectively. The following steps (1 through 2) are performed for each output range step, i, from 1 to n_{rout}.

1. If $rngout_i < r_{pest}$, then $hlim_i$ is set equal to zero. If $rngout_i > r_{pest}$, then the variable, r_o, is set equal to $rngout_i$ and the following ray trace steps a through e are performed until $r_0 \geq r_o$ and $h_0 > ht_{lim}$.

   a. First, the range, r_1, at the end of the ray trace segment is set equal to r_o. Then the current gradient, g_{rd}, is set equal to gr_j. If a_0 is less than zero, then g_{rd} is set equal to gr_{j-1}.

   b. Next, the angle, a_1, at the end of the ray trace segment is found from

   $$a_1 = \mathrm{AP}(a_0, r_1 - r_0) .$$

   c. If a_1 is of the opposite sign as a_0, then a_1 is given by zero and r_1 is given by RP(r_0, a_1-a_0). The variable, h_1, is then given by HP(h_0, a_1, a_0).

   d. Now the value of h_1 is tested. If the value of h_1 is less than or equal to zrt_{j+1} minus 10^{-3}, then h_1 is set equal to zrt_{j+1}, and a_1 and r_1 are re-computed as

   $$a_1 = \sqrt{\mathrm{RADA1}(a_0, h_1 - h_0)} ,$$
   $$r_1 = \mathrm{RP}(r_0, a_1 - a_0),$$
   $$j = j + 1 .$$

   e. The variable, h_0, is then set equal to h_1, r_0 is set equal to r_1, and a_0 is set equal to a_1. Steps a through e are repeated until $r_0 \geq r_o$.

2. Once r_0 has reached r_o, $hlim_i$ is then set equal to h_0. Steps 1 through 2 are repeated for all output range steps.

Tables 32 and 33 identify, describe, and provide units of measure and computational source for each input and output data element of the GETTHMAX SU.

Table 32. GETTHMAX SU input data element requirements.

Name	Description	Units	Source
α_{lim}	Elevation angle of the RO limiting ray	radians	Calling SU
ant_{ref}	Transmitting antenna height relative to h_{minter}	meters	TERINIT SU
f_{MHz}	Frequency	MHz	Calling CSCI
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
gr	Intermediate M-unit gradient array, RO region	(M-unit /m)10^{-6}	REFINIT SU
h_{termax}	Maximum terrain height along profile path	meters	Calling SU
h_{test}	Minimum height at which all trapping refractivity features are below	meters	Calling SU
ht_{lim}	User specified maximum height relative to h_{minter}	meters	TERINIT SU
i_{hybrid}	Integer indicating which submodels will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	N/A	GETMODE SU
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
i_{rtemp}	Temporary number of range steps (used for ray tracing)	N/A	APM_MOD
i_{start}	Array index for height in RO region corresponding to ant_{ref}	N/A	REFINIT SU
i_{tpa}	Number of terrain points in used internally in arrays tx and ty	N/A	APMINIT CSC
λ	Wavelength	meters	APMINIT CSC

Table 32. GETTHMAX SU input data element requirements. (Continued)

Name	Description	Units	Source
ln_{min}	Minimum power of 2 transform size	N/A	APMINIT CSC
n_{rout}	Integer number of output range points desired	N/A	Calling CSCI
r_{adc}	Radians to degrees conversion factor	radians/ degree	APM_MOD
r_{flat}	Maximum range at which the terrain profile remains flat from the source	meters	Calling SU
r_{max}	Maximum output range	meters	Calling CSCI
rngout	Array containing all desired output ranges	meters	APMINIT CSC
slp	Slope of each segment of terrain	N/A	TERINIT SU
tx	Range points of terrain profile	meters	TERINIT SU
ty	Adjusted height points of terrain profile	meters	TERINIT SU
zrt	Height array used for RO calculations	meters	REFINIT SU
z_{test}	Height in PE region that must be reached for hybrid model	meters	Calling SU
z_{tol}	Height tolerance for Newton's method	meters	APMINIT CSC

Table 33. GETTHMAX SU output data element requirements.

Name	Description	Units
a_{launch}	Launch angle used which, when traced, separates PE and XO regions from the RO region	radians
hlim	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	meters
htemp	Heights at which ray is traced to every range in *rtemp*	meters
i_{ap}	Index indicating when the local ray angle becomes positive in array *raya*	N/A
ψ_{lim}	Grazing angle of limiting ray	radians
raya	Array containing all local angles of traced ray a_{launch} at each i_{rtemp} range	radians
r_{pest}	Range at which loss values from the PE model will start being calculated	meters

Table 33. GETTHMAX SU output data element requirements. (Continued)

Name	Description	Units
rtemp	Range steps for tracing to determine maximum PE angle	meters
Θ_{max}	Maximum propagation angle in PE calculations	radians
Θ_{75}	75% of maximum propagation angle in PE calculations	radians
z_{lim}	Maximum height in PE calculation region	meters

5.1.13 Interpolate Profile (INTPROF) SU

The INTPROF SU performs a linear interpolation vertically with height on the refractivity profile, *refref*. Interpolation is performed at each PE mesh height point.

To interpolate vertically at each PE mesh height, the following iteration is performed. The index, *j*, is determined such that for every i^{th} PE bin, ht_i is just greater than $href_j$ and $j < nlvl$. The interpolated profile, *profint*, is then determined from

$$prof\,int_i = refref_{j-1} + con\left(refref_j - refref_{j-1}\right)\frac{ht_i - href_{j-1}}{href_j - href_{j-1}}; \quad for\, i = 1,2,3,\ldots n_{fft},$$

where the array, *ht*, and constant, *con*, have been determined in the APMINIT CSC.

Tables 34 and 35 identify, describe, and provide units of measure and computational source for each input and output data element of the INTPROF SU.

Table 34. INTPROF SU input data element requirements.

Name	Description	Units	Source
con	$10^{-6} k_o$	meters^{-1}	APMINIT CSC
href	Heights of refractivity profile with respect to local ground height	meters	PROFREF SU
ht	PE mesh height array of size n_{fft}	meters	APMINIT CSC
n_{fft}	Transform size	N/A	FFTPAR SU
nlvl	Number of levels in new profile	N/A	PROFREF SU
refref	Refractivity array	M-units	PROFREF SU

Table 35. INTPROF SU output data element requirements.

Name	Description	Units
profint	Profile interpolated to every Δz_{PE} in height	M-units

5.1.14 Free Space Propagator Phase Term (Phase1) SU

The PHASE1 SU initializes the free space propagator array for subsequent use in the PESTEP SU. The propagator term is computed at each PE angle, or p-space, mesh point using the wide-angle propagator. Finally, a filter, or attenuation function (frequently called "window"), is applied to the upper one-quarter of the array corresponding to the highest one-quarter of the maximum propagation angle.

The complex free-space propagator phase array, *frsp*, is given by

$$frsp_j = f_{norm}\left[\mathrm{COS}\left(\Delta r_{PE}\, k_o\left(1-c_j\right)\right) - i\,\mathrm{SIN}\left(\Delta r_{PE}\, k_o\left(1-c_j\right)\right)\right];\ for\ j = 0,1,2,\ldots n_{fft},$$

where i is the imaginary number, $\sqrt{-1}$, f_{norm} is a normalization constant, and c_j is given by

$$c_j = \sqrt{1-\mathrm{AMIN}\left(1,(j\,c_n)^2\right)}\ .$$

Both terms, f_{norm} and c_n, have been previously determined in the APMINIT CSC.

The upper ¼ of the free-space propagator term, *frsp*, is filtered by a cosine-tapered (Tukey) filter array, *filt*, according to

$$frsp_j = filt_{j-n_{3/4}}\ frsp_j;\quad for\ j = n_{3/4}, n_{3/4}+1, n_{3/4}+2, \ldots, n_{fft}\ .$$

Tables 36 and 37 identify, describe, and provide the units of measure and computational source for each input and output data element of the PHASE1 SU.

Table 36. PHASE1 SU input data element requirements.

Name	Description	Units	Source
c_n	Constant equals $\Delta p/k_o$	radians	APMINIT CSC
Δr_{PE}	PE range step	meters	APMINIT CSC
filt	Cosine-tapered (Tukey) filter array	N/A	APMINIT CSC
f_{norm}	Normalization factor	N/A	APMINIT CSC
k_o	Free-space wavenumber	meters^{-1}	APMINIT CSC
n_{fft}	Transform size	N/A	FFTPAR SU
$n_{3/4}$	¾ n_{fft}	N/A	APMINIT CSC

Table 37. PHASE1 SU output data element requirements.

Name	Description	Units
frsp	Complex free space propagator term array	N/A

5.1.15 Environmental Propagator Phase Term (Phase2) SU

The PHASE2 SU calculates the environmental phase term for an interpolated environment profile. This environmental phase term is computed at each PE height, or z-space, mesh point. Finally, a filter, or attenuation function (frequently called "window"), is applied to the upper ¼ of the array corresponding to the highest ¼ of the calculation height domain.

The complex refractivity phase array is given by

$$envpr_j = \text{COS}\left(\Delta r_{PE}\ \ profint_j\right) + i\,\text{SIN}\left(\Delta r_{PE}\ \ profint_j\right);\quad for\ j = 0,1,2,\ldots n_{fft},$$

where i is the imaginary number, $\sqrt{-1}$.

The upper ¼ of *envpr* is filtered by a cosine-tapered (Tukey) filter array, *filt*, according to

$$envpr_j \;=\; filt_{j-n_{3/4}}\ envpr_j;\quad for\ j = n_{3/4}, n_{3/4}+1, n_{3/4}+2, \ldots, n_{fft}\ .$$

Tables 38 and 39 identify, describe, and provide units of measure and computational source for each input and output data element of the PHASE2 SU.

Table 38. PHASE2 SU input data elements requirements.

Name	Description	Units	Source
Δr_{PE}	PE range step	meters	APMINIT CSC
filt	Cosine-tapered (Tukey) filter array	N/A	APMINIT CSC
n_{fft}	Transform size	N/A	FFTPAR SU
$n_{3/4}$	¾ of n_{fft}	N/A	APMINIT CSC
profint	Profile interpolated to every Δz_{PE} in height	M-units	INTPROF SU

Table 39. PHASE2 SU output data element requirements.

Name	Description	Units
envpr	Complex refractivity profile array interpolated every Δz_{PE} in height	N/A

5.1.16 Profile Reference (PROFREF) SU

The PROFREF SU adjusts the current refractivity profile so that it is relative to a reference height, y_{ref}. The reference height is initially the minimum height of the terrain profile. Upon subsequent calls from the PESTEP SU, the refractivity profile is adjusted by the local ground height at each PE range step.

The reference height, y_{ref}, depending on the value of i_{flag}, can be either h_{minter} or the local ground height above h_{minter}. If i_{flag} is zero, the profile arrays, *refref* and *href*, will be relative to h_{minter} and will also be used to initialize *refdum* and *htdum* . If i_{flag} is one, then the profile arrays, *refref* and *href*, will be referenced to the local ground height. The parameter, h_{minter}, is the reference height for internal calculations in the APM CSCI of the complex field, *U*. Both arrays, *refdum* and *htdum*, are dummy arrays containing refractivity values and height values, respectively, for the currently interpolated profile.

The determination of *refref* and *href* proceeds as follows. First, the index, *nlvl*, is initialized to the number of refractivity levels, *lvlep,* in *refdum* and *htdum*, and *refref* and *href* are initialized to zero. Next, a test is made to determine whether the absolute value of the reference height, y_{ref} is greater than 10^{-3} (i.e., is y_{ref} greater than approximately zero). If y_{ref} is approximately zero, the elements of *refref* are set equal to the corresponding M-unit values of *refdum*, and the elements of *href* are set equal to the corresponding height values of *htdum* and the SU is exited.

For the case when y_{ref} is not zero, the following calculations are made. First, the flag i_{bmsl} and the index, j_s, are set equal to zero and minus one, respectively. Then, y_{ref} is tested to determine if it is below mean sea level. If so, i_{bmsl} and j_s are set equal to one and zero, respectively. If y_{ref} is not below mean sea level, then the refractivity profile level at which y_{ref} is just above is determined. The index, j_s, is determined such that $y_{ref} \le htdum_{j_s+1}$ and $y_{ref} > htdum_{j_s}$.

The refractivity at y_{ref} is now computed from

$$r_{mu} = refdum_{j_s} + \left(refdum_{j_s+1} - refdum_{j_s}\right)\frac{y_{ref} - htdum_{j_s}}{htdum_{j_s+1} - htdum_{j_s}}.$$

If y_{ref} falls below mean sea level and the extrapolation flag, i_{extra}, is zero, then r_{mu} is given by

$$r_{mu} = refdum_{j_s} + 0.118\frac{y_{ref} - htdum_{j_s}}{htdum_{j_s+1} - htdum_{j_s}}.$$

The first element in *refref* and *href* is now set equal to r_{mu} and 0, respectively. The number of refractivity levels in the arrays is now $l_{new} = nlvl - j_s$ and the remainder of the current refractivity profile is adjusted in height and stored in *refref* and *href* according to

$$\begin{aligned} refref_j &= refdum_k \\ href_j &= htdum_k - y_{ref}; \quad for\ j = 1,2,3,\ldots l_{new}, \end{aligned}$$

where the index, k, is initialized to j_s+1 at the start and is incremented by one with each iteration of j. The variable, *nlvl*, indicating the number of levels in the newly created profile, is now set to l_{new}.

Next, if i_{flag} equals zero, then *refref* and *href* are used to initialize *refdum* and *htdum* before exiting. Finally, *lvlep* is set equal to *nlvl*.

Tables 40 and 41 identify, describe, and provide the units of measure and computational source for each input and output data element of the PROFREF SU.

Table 40. PROFREF SU input data element requirements.

Name	Description	Units	Source
htdum	Height array for current interpolated profile	meters	REFINTER SU
i_{extra}	Extrapolation flag for refractivity profiles entered below mean sea level i_{extra} = 0; extrapolate to minimum terrain height standard atmosphere gradient i_{extra} = 1; extrapolate to minimum terrain height using first gradient in profile	N/A	Calling CSCI
i_{flag}	Integer flag indicating height at which to reference the refractivity profile i_{flag} = 0; adjust profile relative to h_{minter} i_{flag} = 1; adjust profile relative to local ground height above h_{minter}	N/A	Calling SU
lvlep	Number of height/refractivity levels in profile *refdum* and *htdum*	N/A	Calling CSCI
refdum	M-unit array for current interpolated profile	M-units	REFINTER
y_{ref}	Ground elevation height at current range	meters	Calling SU

Table 41. PROFREF SU output data element requirements.

Name	Description	Units
href	Height array for current interpolated profile	meters
htdum	Dummy array containing height values for current (horizontally interpolated) profile	meters
lvlep	Number of height/refractivity levels in profile	N/A
nlvl	Number of levels in new profile	N/A
refdum	M-unit array for current interpolated profile	M-units
refref	Refractivity array	M-units

5.1.17 Refractivity Initialization (RefInit) SU

The REFINIT SU checks for valid environmental profile inputs and initializes all refractivity arrays used within one application of APM.

Upon entering, the maximum height, h_{large}, at which the refractivity profile is extrapolated, is set to 10^6 meters in a DATA statement. In addition, i_{error} is initialized to zero.

The environmental data are checked to determine if range-dependent profiles have been specified (n_{prof}>1). If so, the range of the last profile entered, $rngprof_{nprof}$, is checked and if it is less than the maximum output range specified, r_{max}, an error message is returned (i.e., i_{error} is set equal to -12) depending on the value of error flag, *lerr*12, set in the TESS-NC CSCI itself. The SU is then exited; otherwise, if no error occurs, the SU proceeds to the next step.

Next, the REFINIT SU tests for valid refractivity level entries for each profile. Every user-specified profile is tested to make sure the first level in the profile begins with a value of zero height (or less than zero if the first level is below mean sea level). If it does not, i_{error} is set to -13 and the SU is exited; otherwise, the SU proceeds to the next step.

A test is then made to determine if the last gradient in each profile is negative. If the last gradient in any profile is negative, i_{error} is set to -14 and the SU is exited; otherwise, an additional refractivity level is extrapolated to height, h_{large}, and added to each profile. The additional level is added according to

$$hmsl_{lvlp,i} = h_{large},$$
$$refmsl_{lvlp,i} = refmsl_{lvlp-1,i} + grd\left[h_{large} - hmsl_{lvlp-1,i}\right]$$

where

$$grd = \frac{refmsl_{lvlp-1,i} - refmsl_{lvlp-2,i}}{hmsl_{lvlp-1,i} - hmsl_{lvlp-2,i}}.$$

The counter for the current profile, i_s, is now initialized to 1 and the range of the next refractivity profile, rv_2, is initialized to $rngprof_{is}$. Next, the results of the extrapolation of the first environmental profile (i.e., the profile at range 0) are transferred to dummy arrays, *htdum* and *refdum*, respectively. The index, *lvlep*, is now set equal to *lvlp*. Duplicate levels in the first profile are removed by a reference to the REMDUP SU, and *refdum* and *htdum* are adjusted to the minimum terrain height by a reference to the PROFREF SU. The parameter, *nlvl*, returned from the PROFREF SU, is now the number of height/refractivity levels in the adjusted *htdum* and *refdum* arrays.

Next, the height and thickness of the highest trapping layer (if one exists), h_{trap} and h_{thick}, respectively, are found relative to h_{minter}. First, h_{trap} and h_{thick} are initialized to zero. Then the following steps (1 through 2) are performed for each i^{th} profile and for each j^{th} refractivity level.

1. The gradient of the current height/refractivity level, *grd*, and its height relative to h_{minter}, h_{p1}, are found from

$$grd = refmsl_{j+1,i} - refmsl_{j,i}$$
$$h_{p1} = hmsl_{j+1,i} - h_{minter}.$$

2. If grd is negative and h_{p1} is greater than h_{trap}, then h_{trap} is set equal to h_{p1}, and h_{p0} and h_{thick} are determined from

$$h_{p0} = hmsl_{j,i} - h_{minter}$$
$$h_{thick} = h_{p1} - h_{p0}$$

Next, the refractivity and height arrays, rm and zrt, respectively, needed in the ray optics (RO) calculations, are built. All elements in zrt are set equal to all elements in $htdum$. An additional height level, equal to ant_{ref}, is included in zrt and the index, i_{start}, is initialized to that height level which corresponds to ant_{ref}. Array rm is given by

$$rm_i = 10^{-6} refdum_i; \qquad for\ i = 0,1,2,\ldots nlvl\ ,$$

with the refractivity level at height, ant_{ref}, interpolated according to

$$rm_{i_{start}} = rm_{i_{start}+1} + \left(ant_{ref} - zrt_{i_{start}-1}\right)\left(\frac{rm_{i_{start}+1} - rm_{i_{start}-1}}{zrt_{i_{start}+1} - zrt_{i_{start}-1}}\right).$$

The total number of levels $levels$ in zrt is reduced by one since the highest level is not needed.

The arrays, gr and q, used in RO and ray-tracing calculations, are determined next. The gradient array gr is given by

$$gr_i = \frac{rm_{i+1} - rm_i}{zrt_{i+1} - zrt_i}; \quad for\ i = 0,1,2,\ldots,levels\ ,$$

where if the absolute value of gr_i is less than 10^{-8}, then gr_i is given by

$$gr_i = 10^{-8}\ \mathrm{SIGN}(1., gr_i)\ .$$

The array q is given by

$$q_i = 2(rm_{i+1} - rm_i); \qquad for\ i = 0,1,2,\ldots,levels\ .$$

Finally, the maximum and minimum M-unit value of the refractivity at range zero, r_{mmax} and r_{mmin}, respectively, are determined as follows. First, r_{mmin} is initialized to rm_0 and r_{mmax} is initialized to r_{mmin}. Then the minimum and maximum values are found for each i^{th} refractivity level from 1 to $nlvl$. The minimum r_{mmin} is found from AMIN(rm_i, r_{mmin}). If rm_i is greater than r_{mmax}, providing i is less than i_{start}, then r_{mmax} is set equal to rm_i. This procedure is repeated for each value of i.

Tables 42 and 43 identify, describe, and provide units of measure and computational source for each input and output data element of the REFINIT SU.

Table 42. REFINIT SU input data element requirements.

Name	Description	Units	Source
ant_{ref}	Transmitting antenna height relative to the reference height h_{minter}	meters	TERINIT SU
h_{minter}	Minimum height of terrain profile	meters	TERINIT SU
$hmsl$	Two-dimensional array containing heights with respect to mean sea level of each profile. Array format must be $hmsl_{i,j}$ = height of i^{th} level of j^{th} profile. $j = 1$ for range-independent cases.	meters	Calling CSCI
$lerr12$	User-provided error flag that will trap on certain errors if set to '.true.'	N/A	Calling CSCI
$lvlp$	Number of height/refractivity levels in profiles	N/A	Calling CSCI
n_{prof}	Number of refractivity profiles	N/A	Calling CSCI
$refmsl$	Two-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be $refmsl_{i,j}$ = M-unit at i^{th} level of j^{th} profile. $j = 1$ for range-independent cases.	M-unit	Calling CSCI
r_{max}	Maximum range	meters	Calling CSCI
$rngprof$	Ranges of each profile. $rngprof_i$ = range of i^{th} profile	meters	Calling CSCI

Table 43. REFINIT SU output data element requirements.

Name	Description	Units
gr	Intermediate M-unit gradient array, RO region	(M-unit /m)10^{-6}
$hmsl$	Two-dimensional array containing heights with respect to mean sea level of each profile. Array format must be $hmsl_{i,j}$ = height of i^{th} level of j^{th} profile. $j = 1$ for range-independent cases	meters
$htdum$	Height array for current interpolated profile	meters
h_{thick}	Thickness of highest trapping layer from all refractivity profiles	meters
h_{trap}	Height of highest trapping layer from all refractivity profiles	meters
i_{error}	Integer value that is returned if any errors exist in input data	N/A
i_s	Counter for current profile	N/A

Table 43. REFINIT SU output data element requirements. (Continued)

Name	Description	Units
i_{start}	RO height index at transmitter	N/A
levels	Number of levels defined in *zrt*, *rm*, *q*, and *gr* arrays	N/A
lvlep	Number of height/refractivity levels in profile, *htdum*, *refdum*	N/A
lvlp	Number of user-specified levels in refractivity profile (for range dependent case all profiles must have same number of levels)	N/A
nlvl	Number of height/refractivity levels in profile, *refref*, *href*	N/A
q	Intermediate M-unit difference array, RO region	2M-unit 10^{-6}
refdum	M-unit array for current profile	M-units
refmsl	Two-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be $refmsl_{i,j}$ = M-unit at i^{th} level of j^{th} profile. $j = 1$ for range-independent cases	M-unit
rm	Intermediate M-unit array, RO region	M 10^{-6}
r_{mmax}	Maximum M-unit value of refractivity profile at range 0	meters
r_{mmin}	Minimum M-unit value of refractivity profile at range 0	meters
rv_2	Range of the next refractivity profile	meters
zrt	Intermediate height array, RO region	meters

5.1.18 Sine Fast-Fourier Transform (SinFFT) SU

A function with a common period, such as a solution to the wave equation, may be represented by a series consisting of sines and cosines. This representation is known as a Fourier series. An analytical transformation of the function, known as a Fourier transform, may be used to obtain a solution for the function.

The solution to the PE approximation to Maxwell's wave equation is obtained by using such a Fourier transformation function. The APM CSCI uses only the real-valued sine transformation in which the real and imaginary parts of the PE equation are transformed separately. The Fourier transformation provided with the APM CSCI is described by Bergland (1969) and Cooley (1970). The FORTRAN source code is listed in Appendix A.

Other sine fast Fourier transform (FFT) routines are available in the commercial market, and such a sine FFT may already be available within another TESS-NC CSCI. The selection of which FFT ultimately used by the APM CSCI is left to the application designer as every sine FFT will have hardware and/or software performance impacts. For this reason, it is be-

yond the scope of this document to describe the numerical implementation of the FFT algorithm.

Tables 44 and 45 identify, describe, and provide the units of measure and computational source for each input and output data element of the SINFFT SU.

Table 44. SINFFT input data element requirements.

Name	Description	Units	Source
n_{fft}	Transform size	N/A	FFTPAR SU
x	Field array to be transformed—dimensioned $2^{n_{fft}}$ in calling SU	µV/m	FFT SU

Table 45. SINFFT output data element requirement.

Name	Description	Units
x	Sine transform of field	

5.1.19 Terrain Initialization (TERINIT) SU

The TERINIT SU examines and initializes terrain arrays for subsequent use in PE calculations. It tests for and determines a range increment if it is found that range/height points are provided in fixed range increments. The minimum terrain height is determined, and the entire terrain profile is adjusted in height so that all internal calculations are referenced to this height. This is done to maximize the PE transform calculation volume.

First, several variables are initialized. The logical flag, f_{ter}, used to indicate whether the application at hand is a terrain case, is set equal to '.false.' The integer flag, i_{error}, that is returned if any errors exist in input data, is set equal to zero. The maximum tangent ray angle, α_u, from source to terrain peak along the profile path is set equal to zero. The minimum height of the terrain profile, h_{minter}, is set equal to zero. The transmitting antenna height, ant_{ref}, relative to the reference height, h_{minter}, is set equal to ant_{ht}, the transmitting antenna height above the local ground at range zero. The maximum terrain height, h_{termax}, along the profile path is set equal to zero. Finally, if the number of terrain points, i_{tp}, specified is greater than zero, then f_{ter} is set equal to '.true.'.

If performing a terrain case (f_{ter} = '.true.'), the following steps (1 through 10) are performed, otherwise, the SU proceeds to step 10.

1. First, all terrain range points are checked in array *terx* to ensure they are steadily increasing. If they are not, the error flag i_{error} is set equal to -17 and the SU is exited. Otherwise, the SU proceeds to step 2.

2. Next, a test is made to determine whether the first range value is zero. If it is not, the error flag, i_{error}, is set equal to -18 and the SU is exited; otherwise, the SU proceeds to step 3.

3. A check is now made to determine if the specified terrain range points are spaced at fixed increments. In this procedure, three variables, $rdif_1$, r_{frac}, and r_{difsum} are initialized to $terx_2$-$terx_1$, zero, and $rdif_1$, respectively. The variable, $rdif_1$, is the difference between adjacent terrain point ranges. The variable, r_{frac}, is the ratio between adjacent terrain point differences. The variable, r_{difsum}, is the running sum of adjacent terrain point differences. The final value for r_{difsum} and maximum, r_{frac}, are determined as

$$rdif_2 = \mathrm{AMAX}\left(10^{-3}, terx_{i+1} - terx_i\right); \quad for\ i = 2,3,4,\ldots,i_{tp} - 1$$

$$r_{frac} = \frac{rdif_2}{rdif_1},$$

$$r_{difsum} = r_{difsum} + rdif_2,$$

where $rdif_1$ is set equal to the previous value of $rdif_2$ before each subsequent calculation of a new $rdif_2$, and r_{frac} is the maximum of all ratios computed.

4. If it is determined that the terrain points are spaced at fixed range increments, then the range spacing r_{fix} is set to this increment. Assuming that the range points are not equally spaced, r_{fix} is initially set equal to zero. If the value of r_{frac} is less than 1.05, then r_{fix} is determined from

$$r_{fix} = \mathrm{NINT}\left(\frac{r_{difsum}}{i_{tp}-1}\right).$$

5. Next a test is made to see if the last range point in the profile meets or exceeds the maximum output range, r_{max}. If the logical flag, *lerr*6, is '.true.', then trapping for the condition, $terx_{itp} < r_{max}$, occurs. If this condition occurs, the flag, i_{error}, is set equal to -6 and the SU is exited; otherwise, the SU proceeds to step 6.

6. The minimum height of the terrain profile is found by initially setting h_{minter} equal to h_{max}. The minimum height, h_{minter}, is now determined from the minimum of h_{minter} and $tery_i$ in an iterative loop for all terrain points for index i running from 1 to i_{tp}.

7. Now the entire terrain profile is adjusted by h_{minter} such that this is the new zero reference. The adjusted terrain profile is stored in the arrays, *tx* and *ty*. The maximum height of the terrain h_{termax} is also obtained from *ty* in the same manner as h_{minter} is determined in step 6.

8. An extra point is added to the arrays, *tx* and *ty*. If tx_{itp} is less than r_{max}, then tx_{itpa} is set equal to r_{max} times 1.1. The input index, i_{tpa}, is the number of terrain points used internally in the arrays, *tx* and *ty*. If tx_{itp} is greater or equal to r_{max}, then tx_{itpa} is set equal to tx_{itp} times 1.1. Finally, the array element ty_{itpa} is set equal to ty_{itp}. If h_{max} does not exceed the maximum height of the terrain profile, h_{termax}, then the error flag, i_{error}, is set equal to -8, and the SU is exited; otherwise, the SU proceeds with step 9.

9. The variable, ant_{ref}, is set equal to ant_{ht} plus ty_1. Next, the array of terrain slopes, *slp,* and the maximum tangent ray angle, α_u, from the source to the terrain peak along the profile path are found as follows. The slope, slp_i, for each i^{th} terrain segment is given by

$$slp_i = \frac{ty_{i+1} - ty_i}{\mathrm{AMAX}\left(tx_{i+1} - tx_i, 10^{-5}\right)}; \quad for\, i = 1,2,3,\ldots i_{tpa} - 1.$$

If the value of ty_i is greater than ant_{ref}, then the maximum tangent angle, α_u, from the source to each terrain point is calculated as

$$angle = \mathrm{ATAN}\left(\frac{ty_i - ant_{ref}}{tx_i}\right); \quad for\, i = 1,2,3,\ldots i_{tpa} - 1,$$

$$\alpha_u = \mathrm{AMAX}(angle, \alpha_u)$$

After α_u is determined, 0.5° is added to its value.

10. Before exiting, the minimum height, hm_{ref}, relative to h_{minter}, is found from the difference between the minimum specified output height, h_{mi} and h_{minter}. The maximum height limit, ht_{lim}, relative to h_{minter}, is given by the difference between h_{max} and h_{minter}. If the antenna height, ant_{ref}, is greater than ht_{lim}, the error code, i_{error}, is set to -9.

Tables 46 and 47 identify, describe, and provide units of measure and computational source for each input and output data element of the TERINIT SU.

Table 46. TERINIT SU input data element requirements.

Name	Description	Units	Source
ant_{ht}	Transmitting antenna height above local ground	meters	Calling CSCI
h_{max}	Maximum output height with respect to mean sea level	meters	Calling CSCI
h_{min}	Minimum output height with respect to mean sea level	meters	Calling CSCI
i_{tp}	Number of height/range points in profile	N/A	Calling CSCI
i_{tpa}	Number of height/range points pairs in profile, tx, ty	N/A	APMINIT CSC
lerr6	User-provided error flag that will trap on certain errors if set to '.true.'	N/A	Calling CSCI
r_{max}	Maximum output range	meters	Calling CSCI
terx	Range points of terrain profile	meters	Calling CSCI
tery	Height points of terrain profile	meters	Calling CSCI

Table 47. TERINIT SU output data element requirements.

Name	Description	Units
α_u	Maximum tangent ray angle from the source to the terrain peak along profile height	radians
ant_{ref}	Transmitting antenna height relative to the reference height, h_{minter}	meters
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A
h_{minter}	Minimum height of terrain profile	meters
hm_{ref}	Height relative to h_{minter}	meters
ht_{lim}	User-supplied maximum height relative to h_{minter} (*i.e.*, $ht_{lim} = h_{max} - h_{minter}$)	meters
h_{termax}	Maximum terrain height along profile path	meters
i_{error}	Integer value that is returned if errors exist in input data	N/A
r_{fix}	Fixed range increment of terrain profile	meters
slp	Slope of each segment of terrain	N/A
tx	Range points of terrain profile	meters
ty	Adjusted height points of terrain profile	meters

5.1.20 Troposcatter Initialization (TROPOINIT) SU

The TROPOINIT SU initializes all variables and arrays needed for subsequent troposcatter calculations. The tangent range and tangent angle are determined from the source and the tangent range and tangent angles are determined for all receiver heights and stored in arrays.

First, several variables are initialized. The first of these, the surface refractivity, sn_{ref}, is set equal to $refdum_0$, the first element of the dummy array containing M-unit values for the current (interpolated) refractivity profile taken relative to h_{minter}. Then, the array, $\vartheta 0$, containing angles used in determining the common volume scattering angle, is found from

$$\vartheta 0_i = \frac{rngout_i}{a_{ek}}; \quad for\ i = 1,2,3,\ldots,n_{rout},$$

where a_{ek} is $^4/_3$ times the Earth's mean radius. A term used in the troposcatter transmission loss calculation, sn_1, is determined from

$$sn_1 = 0.031 - 0.00232\, sn_{ref} + 5.67 \times 10^{-6}\, sn_{ref}^2 .$$

A constant needed in the troposcatter calculation, r_f, is determined from 0.0419 times the frequency, f_{MHz}. A second constant needed in the troposcatter calculation, rt_1, is found from r_f times the adjusted transmitting antenna height, ant_{ref}. Next, the tangent angle from the source, ϑ_{1s}, for smooth surface is computed from

$$\vartheta_{1s} = -\frac{d_{1s}}{a_{ek}};$$

$$d_{1s} = \sqrt{2a_{ek} ant_{ref}},$$

where d_{1s} is the tangent range from the source for smooth surface. The variable α_{ld} is determined from

$$\alpha_{ld} = 20\,\mathrm{LOG}\left[f(\alpha_d)\right],$$

$$\alpha_d = \vartheta_{1s} + 10^{-6},$$

where α_d represents the lowest direct ray angle in the RO region, and $f(\alpha_d)$ is the antenna pattern factor, obtained from referencing the ANTPAT SU, for the direct angle.

The minimum range, r_{hor1}, at which the diffraction field solutions are applicable and the intermediate region ends is determined for smooth surface and zero receiver height. The variable, r_{hor1}, is given by

$$r_{hor1} = \sqrt{2a_{ek}\, ant_{ref}} + 230200.0\left(\frac{e_k^{\,2}}{f_{MHz}}\right)^{0.333},$$

where e_k is the effective Earth's radius factor value 1.3333333.

Next, the tangent ranges and angles for all output receiver heights are computed and stored in the arrays, $d2s$ and $\vartheta 2s$, respectively. The minimum ranges at which diffraction field solutions are applicable (for smooth surface) for all output receiver heights are determined and stored in the array, rdt. Height differences between ant_{ref} and each output receiver height are also computed and stored in $adif$. These arrays are given by

$$d2s_i = \sqrt{2a_{ek}\, zout_i},$$

$$\vartheta 2s_i = -\frac{d2s_i}{a_{ek}},$$

$$rdt_i = r_{hor1} + d2s_i$$

$$adif_i = ant_{ref} - zout_i,$$

where the computation is performed for each i^{th} output receiver height, $zout_i$, provided $zout_i$ is greater than or equal to 0, and i ranges from 1 to n_{zout}.

If f_{ter} is '.true.', the following steps (1 through 4) are performed to compute all steadily increasing tangent ranges and angles from the source, $ad1$ and $\vartheta 1t$, respectively.

1. First, α_{ld} and the index j are each set equal to zero. The current largest tangent angle, t_{st}, from the source, is initialized to ϑ_{1s}.
2. The following steps (a through c) are performed for each i^{th} terrain point from 1 to i_{tpa}.
   a. The tangent angle at each terrain point is given by

$$\alpha_1 = \frac{ty_i - ant_{ref}}{tx_i} - \frac{tx_i}{2a_{ek}} .$$

   b. If α_1 is greater than t_{st}, then if tx_i is greater than d_{1s}, and simultaneously j is equal to 0, then j is incremented by 1, $\vartheta 1t_j$ is set equal to ϑ_{1s}, and $ad1_j$ is set equal to d_{1s}.
   c. If α_1 is greater than t_{st}, j is incremented by 1, $\vartheta 1t_j$ is set equal to α_1, and $ad1_j$ is set equal to tx_i. The variable t_{st} is now set equal to α_1 and steps 2a through 2b are repeated for all terrain range/height pairs.
3. If no tangent angles or ranges have been found to satisfy conditions in step 2b and 2c above (i.e., if j is still 0 after all iterations in steps 2a through 2c), then j is incremented by 1, $\vartheta 1t_j$ is set equal to ϑ_{1s}, and $ad1_j$ is set equal to d_{1s}.
4. All index counters, ktr_1, j_{t1}, and j_{t2} (used in the TROPO SU) are initialized to j, 1, and 1, respectively.

Finally, the troposcatter loss term, $tlst_s$, is given by

$$tlst_s = 54.9 + 30.0\,\mathrm{LOG}\left(f_{MHz}\right) - 0.2\,sn_{ref} - \alpha_{ld} .$$

Tables 48 and 49 identify, describe, and provide units of measure and computational source for each input and output data element of the TROPOINIT SU.

Table 48. TROPOINIT SU input data element requirements.

Name	Description	Units	Source
a_{ek}	$^4/_3$ effective earth's radius	meters	APM_MOD
a_{ek2}	Twice $^4/_3$ effective earth's radius	meters	APMINIT CSC
ant_{ref}	Transmitting antenna height relative to h_{minter}	meters	TERINIT SU
e_k	$^4/_3$ effective earth's radius factor	N/A	APM_MOD
f_{MHz}	Frequency	MHz	Calling CSCI
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU

Table 48. TROPOINIT SU input data element requirements.

Name	Description	Units	Source
i_{tpa}	Number of height/range points pairs in profile, tx, ty	N/A	APMINIT CSC
n_{rout}	Integer number of output range points desired	N/A	Calling CSCI
n_{zout}	Integer number of output height points desired	N/A	Calling CSCI
refdum	M-unit array for current interpolated profile	M-units	REFINTER SU
rngout	Array containing all desired output ranges	meters	APMINIT CSC
tx	Range points of terrain profile	meters	TERINIT SU
ty	Adjusted height points of terrain profile	meters	TERINIT SU
zout	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 49. TROPOINIT SU output data element requirements.

Name	Description	Units
$ad1$	Array of tangent ranges from source height—used with terrain profile	meters
adif	Height differences between ant_{ref} and all output receiver heights	meters
$d2s$	Array of tangent ranges for all output receiver heights over smooth surface	meters
j_{t1}	Index counter for $ad1$ and $\vartheta 1t$ arrays	N/A
j_{t2}	Index counter for tx and ty arrays indicating location of receiver range	N/A
ktr_1	Number of tangent ranges from source height	N/A
rdt	Array of minimum ranges at which diffraction field solutions are applicable (for smooth surface) for all output receiver heights.	meters
r_f	Constant used for troposcatter calculations	meters^{-1}
rt_1	$r_f * ant_{ref}$	N/A
sn_1	Term used in troposcatter loss calculation	N/A
$\vartheta 0$	Array of angles used to determine common volume scattering angle	radians
ϑ_{1s}	Tangent angle from source (for smooth surface)	radians

Table 49. TROPOINIT SU output data element requirements.

Name	Description	Units
$\vartheta 1t$	Array of tangent angles from source height—used with terrain profile	radians
$\vartheta 2s$	Array of tangent angles from all output receiver heights —used with smooth surface	radians
$tlst_s$	Troposcatter loss term for smooth surface case	dB

5.1.21 Starter Field Initialization (XYINIT) SU

The XYINIT SU calculates the complex PE solution at range zero.

Upon entering this SU, several constant terms that will be employed over the entire PE mesh are calculated. The PE mesh is defined by the number of points in the mesh, n_{fft}, and by the mesh size, Δp. The constant terms include: (1) the angle difference between mesh points in p-space, $\Delta\Theta$; (2) a height-gain value at the source (transmitter), ant_{ko} ; and (3) the normalization factor, s_{gain} , used in the determination of the complex array containing the field, U. The normalization factor, s_{gain} , is given by

$$s_{gain} = \frac{\sqrt{\lambda}}{z_{max}} .$$

The angle difference between mesh points in p-space, $\Delta\Theta$, is given by

$$\Delta\Theta = \frac{\Delta p}{k_o},$$

where k_o is the free-space wave number. The height-gain value at the source (transmitter), ant_{k_o} , is given by

$$ant_{k_o} = k_o \, ant_{ht} ,$$

where ant_{ht} is the transmitting antenna height above the local ground in meters.

For each point in the PE p-space mesh (i.e., $i = 0$ to n_{fft}), the following steps are performed. First, the sine of the direct-path ray elevation angle , p_k , is determined from

$$p_k = i \ \Delta\Theta .$$

Next, the direct ray elevation angle, α_d, is set equal to $\mathrm{SIN}^{-1}(p_k)$ and the antenna pattern factors, $f(\alpha_d)$ for the direct path and $f(-\alpha_d)$ for the reflected path, are determined by referencing the ANTPAT SU. Then, the complex portions of the PE solution, U, are determined from the antenna pattern factors, elevation angle, and normalization factor from

$$U_i = s_{gain} \left[f(\alpha_d) D_{term} - f(-\alpha_d) R_{term} \right]$$

where the field, R_{term}, due to an image point source at height, ant_{ht}, is given by

$$R_{term} = \text{COS}\left(p_k\, ant_{k_o}\right) + j\,\text{SIN}\left(p_k\, ant_{k_o}\right)$$

and the field, D_{term}, due to a real point source at the height ant_{ht} is given by

$$D_{term} = \text{COS}\left(p_k\, ant_{k_o}\right) - j\,\text{SIN}\left(p_k\, ant_{k_o}\right).$$

In the above two equations, j is the imaginary number, $\sqrt{-1}$.

Finally, the upper ¼ of the field values are filtered. A cosine-tapered (Tukey) filter array, *filt*, is used for this purpose. The initial PE field U is given by

$$U_i = U_i\, filt_{i-n_{3/4}};\quad for\ i = n_{3/4}, n_{3/4}+1, n_{3/4}+2, \ldots, n_{fft},$$

where $n_{3/4}$ is equal to ¾ of n_{fft} .

Tables 50 and 51 identify, describe, and provide units of measure and computational source for each input and output data element of the XYINIT SU.

Table 50. XYINIT SU input data element requirements.

Name	Description	Units	Source
ant_{ht}	Transmitting antenna height above local ground	meters	Calling CSCI
Δp	Mesh size in angle- (or p-) space	radians	APMINIT CSC
filt	Cosine-tapered (Tukey) filter array	N/A	APMINIT CSC
k_o	Free-space wave number	meters^{-1}	APMINIT CSC
λ	Wavelength	meters	APMINIT CSC
n_{fft}	Transform size	N/A	FFTPAR SU
$n_{3/4}$	¾ n_{fft}	N/A	APMINIT CSC
z_{max}	Total height of the FFT/PE calculation domain	meters	FFTPAR SU

Table 51. XYINIT SU output data element requirements.

Name	Description	Units
U	Transform of complex field	

5.2 ADVANCED PROPAGATION MODEL STEP (APMSTEP) CSC

The APMSTEP SU advances the entire APM CSCI algorithm one output range step, referencing various SUs to calculate the propagation loss at the current output range.

Upon entering the APMSTEP SU, the current output range, r_{out}, and the square of the output range, r_{sq}, are updated, and all *mloss* array integer indices for the various calculation regions are initialized. The PESTEP SU is then referenced to determine all propagation loss values within the PE calculation region. The variable, *mloss*, is returned with integer indices, j_{ps} and j_{pe}, corresponding to the start and end, respectively, of propagation loss values within *mloss*.

If APM is executing under the full hybrid mode ($i_{hybrid} = 1$) and the current output range is less than the range at which the XO region begins ($r_{out} < r_{atz}$), the following steps 1 and 2 are performed.

1. The starting and ending *mloss* array indices for FE calculations, j_{fs} and j_{fe}, respectively, are determined. For ranges less than 2.5 km, j_{fs} is set equal to 0 for vertical polarization, or 1 for horizontal polarization, and j_{fe} is set equal to n_{zout}. For ranges greater than 2.5 km, j_{fs} is set equal to the maximum of j_{pe}+1, or 1 greater than the output height index which corresponds to the height just above the FE 5° angle limit. The FEM SU is then referenced and propagation loss values within the FE region are computed and returned in *mloss*.

2. If the current output range is greater than 2.5 km, then the starting and ending *mloss* array indices for RO calculations, j_{rs} and j_{re}, respectively, are determined. These indices are based on the values of j_{ps}, j_{pe}, j_{fs}, and j_{fe} such that at every range step, j_{rs} will always be greater than the ending index of the PE region, (j_{pe}), and j_{re} will be less than the starting index of the FE region, (j_{fs}). The ROM SU is then referenced and propagation loss values within the RO region are computed and returned in *mloss*.

Once all necessary propagation loss values have been computed for a particular output range, the gaseous absorption calculation flag, k_{abs}, is tested and if it is greater than 0, then loss in centibels due to gaseous absorption, l_{abscb}, is computed as follows:

$$l_{abscb} = \mathrm{NINT}(r_{out}\ gas_{att})$$

where gas_{att} is the gaseous absorption attenuation rate in dB/km. The absorption loss is then added to all loss values in *mloss*.

Upon exiting, if the final output range step has been reached, the integer counters, j_{t1} and j_{t2} associated with troposcatter calculations are initialized to 1.

Tables 52 and 53 identify, describe, and provide units of measure and computational source for each input and output data element of the APMSTEP SU.

Table 52. APMSTEP CSC input data element requirements.

Name	Description	Units	Source
gas_{att}	Gaseous absorption attenuation rate	dB/km	GASABS SU
htfe	Array containing the height at each output range separating the FE region from the RO region (full hybrid mode), or the FE region from the PE region (partial hybrid mode)	meters	FILLHT SU
ht_{lim}	Maximum height relative to h_{minter}	meters	TERINIT SU
i_{hybrid}	Integer indicating which submodels will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	N/A	GETMODE SU
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
i_{stp}	Current output range step index	N/A	Calling CSCI
i_{zg}	Number of output height points corresponding to local ground height at current output range, r_{out}	N/A	CALCLOS SU
k_{abs}	Gaseous absorption calculation flag: k_{abs} = 0; no absorption loss k_{abs} = 1; compute absorption loss based on air temperature, t_{air}, and absolute humidity, abs_{hum} k_{abs} = 2; compute absorption loss based on specified absorption attenuation rate, γ_a	N/A	APMINIT CSC
n_{rout}	Integer number of output range points desired	N/A	Calling CSCI
n_{zout}	Integer number of output height points desired	N/A	Calling CSCI
r_{atz}	Range at which z_{lim} is reached (used for hybrid model)	meters	APMINIT CSC
rngout	Array containing all desired output ranges	meters	APMINIT CSC
rsqrd	Array containing the square of all desired output ranges	meters^2	APMINIT CSC
r_{tst}	Range at which to begin RO calculations (equal to 2.5 km)	meters	APM_MOD
zout	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 53. APMSTEP CSC output data element requirements.

Name	Description	Units
j_{end}	Index at which valid loss values in *mloss* end	N/A
j_{start}	Index at which valid loss values in *mloss* start	N/A
j_{t1}	Index counter for *adl* and ϑlt arrays	N/A
j_{t2}	Index counter for *tx* and *ty* arrays indicating location of receiver range	N/A
mloss	Propagation loss array	cB
r_{out}	Current desired output range	meters

5.2.1 Calculate Propagation Loss (CALCLOS SU)

The CALCLOS SU determines the propagation loss at each output height point at the current output range.

At the outset a minimum propagation factor, $pfac_{min}$, is set to 300 dB.

Then an in-line function, PLINT, for linear interpolation between two values, pl_1 and pl_2, is defined by

$$\mathrm{PLINT}\left(pl_1,\ pl_2,\ f_{rac} \right) \ = \ pl_1 \ + \ f_{rac} \left(pl_2 - pl_1 \right),$$

where f_{rac} is the fractional distance from pl_1 to pl_2 for which the interpolation is being made.

Several variables are initialized. The output range, r_{out}, is updated based on the current range step, i_{stp}. The height of the terrain at the current and last ranges, y_{ch} and y_{lh}, respectively, are determined relative to the reference height, hm_{ref}.

Next, the interpolated ground height, z_{int}, at the current output range and the number of vertical output points, i_{zg}, that correspond to this ground height are determined. First, the interpolated ground height is given by

$$z_{int} \ = \ \mathrm{PLINT}\left(y_{last}\ ,\ y_{cur}\ ,\ xx \right),$$

where the parameter xx is given in terms of the PE range step, Δr_{PE}, by

$$xx = \frac{r_{out} - r_{last}}{\Delta r_{PE}}.$$

Having determined z_{int}, i_{zg} is then computed from

$$i_{zg} \ = \ \mathrm{INT}\left(\frac{z_{int} - hm_{ref}}{\Delta z_{out}} \right),$$

where Δz_{out} is the output height increment. Next, all elements in array *mloss* from 1 to i_{zg} are set to zero, and the index j_{start}, representing beginning valid loss values in the *mloss* array, is set to the maximum of 0 or i_{zg}. If vertical polarization is used, then 1 is added to the value of j_{start}.

If the current output range is greater than the range, r_{pest}, at which PE solutions are valid, then the calculation of loss values begins. If this condition is not satisfied, then the *mloss* array is set to -1 for values of the array index from j_{start} up to and including the number of output height points desired, (n_{zout}), and the SU is exited.

Once it is determined that loss calculations will be performed, several parameters are computed. If the logical variable f_{ter} is '.true.', then a terrain case is being performed. The two indices, i_{p1} and i_{p2}, are given by

$$i_{p1} = \mathrm{AMAX}\left(0, \mathrm{INT}\left\{\frac{y_{lh}}{\Delta z_{out}}\right\}\right)$$

and

$$i_{p2} = \mathrm{AMAX}\left(0, \mathrm{INT}\left\{\frac{y_{ch}}{\Delta z_{out}}\right\}\right).$$

These indices indicate the first output height point in array, *zout*, where propagation loss will be computed at the last and current PE ranges. Next, the output heights, $zout_{ip_1}$ and $zout_{ip_2}$, relative to y_{last} and y_{cur}, respectively, are checked to ensure they are positive. If not, the two indices, i_{p1} and i_{p2}, are incremented by a value of 1. For values of the array index from 0 up to and including i_{p1}, the array of propagation factors, *rfac*1, at valid height points for range, r_{last}, are set to the minimum propagation factor, $pfac_{min}$, for later interpolation. For values of the array index from 0 up to and including i_{p2}, the array of propagation factors, *rfac*2, at valid height points for range, $r_{out,}$ are set to the minimum propagation factor, $pfac_{min}$, for later interpolation.

If the logical variable f_{ter} is '.false.' (i.e., a smooth surface case), then both i_{p1} and i_{p2} are set to 0.

Next, the height/integer value, j_{end}, indicating the end of valid loss values, is determined as

$$j_{end} = \mathrm{AMAX}\left(0, \mathrm{NINT}\left\{\frac{\mathrm{AMIN}\left[z_{lim}, \mathrm{AMAX}\left(z_{int}, hlim_{i_{stp}}\right)\right] - hm_{ref}}{\Delta z_{out}}\right\}\right),$$

where i_{stp} is the current output range step, and $hlim_{i_{stp}}$ is the height at the current output range step separating the PE region from the FE, RO, or XO regions. Note that for terrain cases, ray tracing was performed using the direct ray angle and sometimes $hlim_{i_{stp}}$ may be less than the local ground height. In that case, this SU exits from the propagation loss calculation.

The propagation loss values are determined from the propagation factors, $rfac1_i$ and $rfac2_i$, and from the parameter, *xx*, defined earlier in this section. If r_{loglst} $(10\,\mathrm{LOG}(r_{last}))$ is greater than zero (it is initialized to 0 for i_{stp}=1), then the GETPFAC SU is referenced to determine the propagation factor, $rfac1_i$, which is given by

$$rfac1_i = \text{GETPFAC}\left(Ulast, r_{loglst}, zout_i - y_{last}\right); \qquad for \;\; i = i_{p1}, i_{p1}+1, i_{p1}+2, \ldots, j_{end},$$

where *Ulast* is the complex field array at the previous PE range. Next, the propagation factor, $rfac2_i$, is given by

$$rfac2_i = \text{GETPFAC}\left(U, r_{log}, zout_i - y_{cur}\right); \qquad for \;\; i = i_{p2}, i_{p2}+1, i_{p2}+2, \ldots, j_{end},$$

where U is the complex field array at the current PE range, and r_{log} is $10\,\text{LOG}(r)$.

Next, if using the PE model only or the partial hybrid mode (PE & XO models), heights corresponding to the area outside the valid PE calculation region are determined and propagation loss is set equal to -1 within *mloss* for those heights. If using the full or partial hybrid modes, the propagation factor at the last PE height point is determined at both the previous and current PE ranges. Linear interpolation is then performed, via the PLINT in-line function, to compute the propagation loss at range, r_{out}, and height, z_{int}. The loss and height are then stored in array, *ffrout*, for subsequent interpolation in the EXTO SU.

Next, the propagation loss at range, r_{out}, is found by interpolating between the current and previous PE ranges. Again referencing the in-line function PLINT, the propagation loss, $rloss_k$, is given by

$$rloss_k = \text{PLINT}\left[rfac1_k, rfac2_k, xx\right] + fslr_{i_{stp}}; \quad for \;\; k = j_{start}, j_{start}+1, j_{start}+2, \ldots, j_{end},$$

where $fslr_{i_{stp}}$ is the free-space loss in dB at range r_{out}.

If the troposcatter calculation flag, i_{tropo}, is 1, then the TROPO SU is referenced to compute troposcatter loss from height $zout_{j_{start}}$ to $zout_{j_{end}}$, and this is added, if necessary, to *rloss*.

Finally, the loss in centibels is given by

$$mloss_k = \text{NINT}\left[10\;\text{LOG}\left(rloss_k\right)\right]; \qquad for \;\; k = j_{start}, j_{start}+1, j_{start}+2, \ldots, j_{end},$$

with the remaining elements in *mloss* set equal to -1 (i.e., $mloss_k$=-1 for k=j_{end}+1 to n_{zout}) before exiting.

Tables 54 and 55 identify, describe, and provide units of measure and computational source for each input and output data element of the CALCLOS SU.

Table 54. CALCLOS SU input data element requirements.

Name	Description	Units	Source
Δr_{PE}	PE range step	meters	APMINIT CSC
Δz_{out}	Output height increment	meters	APMINIT CSC
Δz_{PE}	PE mesh height increment (bin width in z-space)	meters	FFTPAR SU
$fslr$	Free-space loss array for output ranges	dB	APMINIT CSC

Table 54. CALCLOS SU input data element requirements. (Continued)

Name	Description	Units	Source
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
$hlim$	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	meters	GETTHMAX SU
hm_{ref}	Height relative to h_{minter}	meters	TERINIT SU
$htfe$	Array containing the height at each output range separating the FE region from the RO region (full hybrid mode), or the FE region from the PE region (partial hybrid mode)	meters	FILLHT SU
i_{hybrid}	Integer indicating which submodels will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	N/A	GETMODE SU
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
i_{stp}	Current output range step index	N/A	Calling SU
i_{tropo}	Troposcatter calculation flag: i_{tropo} = 0; no troposcatter calcs i_{tropo} = 1; troposcatter calcs	N/A	Calling CSCI
i_{xo}	Number of range steps in XO calculation region.	N/A	APMINIT CSC
n_{zout}	Integer number of output height points desired	N/A	Calling CSCI
r_{last}	Previous PE range	meters	Calling SU
r_{log}	10 LOG(PE range, r)	N/A	PESTEP SU
r_{loglst}	10 LOG(previous PE range, r_{last})	N/A	PESTEP SU
$rngout$	Array containing all desired output ranges	meters	APMINIT CSC

Table 54. CALCLOS SU input data element requirements. (Continued)

Name	Description	Units	Source
r_{pest}	Range at which PE loss values will start being calculated	meters	GETTHMAX SU
U	Complex field at current PE range, r	µV/m	PESTEP SU
$Ulast$	Complex field at previous PE range, r_{last}	µV/m	PESTEP SU
y_{cur}	Height of ground at current range, r	meters	PESTEP SU
y_{last}	Height of ground at previous range, r_{last}	meters	PESTEP SU
z_{lim}	Height limit for PE calculation region	meters	GETTHMAX SU
$zout$	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 55. CALCLOS SU output data element requirements.

Name	Description	Units
$ffrout$	Array of propagation factors at each output range beyond r_{atz} and at height, z_{lim}	dB
i_{zg}	Number of output height points corresponding to local ground height at current output range, r_{out}	N/A
j_{end}	Index at which valid loss values in *mloss* end	N/A
j_{start}	Index at which valid loss values in *mloss* begin	N/A
$mloss$	Propagation loss	cB
$rfac1$	Propagation factor at valid output height points from PE field at range, r_{last}.	dB
$rfac2$	Propagation factor at valid output height points from PE field at range, r	dB
$rloss$	Propagation loss	dB

5.2.2 DOSHIFT SU

The DOSHIFT SU shifts the field by the number of bins or PE mesh heights corresponding to the local ground height.

Upon entry, the number of bins to be shifted is determined. First, the difference, y_{diff}, between the height of the ground, y_{last}, at the previous range and that at the current PE range, y_{cur}, is determined from

$$y_{diff} = y_{cur} - y_{last}.$$

The number of bins to be shifted, k_{bin}, is found from

$$k_{bin} = \mathrm{NINT}\left(\frac{|y_{diff}|}{\Delta z_{PE}} \right) .$$

The PE solution U is then shifted downward if the local ground is currently at a positive slope ($y_{diff} > 0$), upward if the local ground is at a negative slope ($y_{diff} < 0$) and otherwise not shifted. When the PE solution has been shifted down, the value of the PE solution, U, for the upper k_{bin} elements are set to zero. Likewise, when the PE solution has been shifted upwards, the lower k_{bin} elements are set to zero.

Tables 56 and 57 identify, describe, and provide units of measure and computational source for each input and output data element of the DOSHIFT SU.

Table 56. DOSHIFT SU input data element requirements.

Name	Description	Units	Source
Δz_{PE}	PE mesh height increment (bin width in z-space)	meters	FFTPAR SU
n_{fft}	Transform size	N/A	FFTPAR SU
n_{m1}	$n_{fft} - 1$	N/A	APMINIT CSC
U	Complex field at range, r	μV/m	Calling SU
y_{cur}	Height of ground at current range, r	meters	Calling SU
y_{last}	Height of ground at previous range, r_{last}	meters	Calling SU

Table 57. DOSHIFT SU output data element requirements.

Name	Description	Units
U	Complex field after bin shift	μV/m

5.2.3 Flat Earth Model (FEM) SU

The FEM SU computes propagation loss at a specified range based upon flat-earth approximations. The following steps (1 through 10) are performed for each APM height output point from j_{fs} to j_{fe}.

1. The receiver height at the j^{th} output point, $zout_j$, is first adjusted relative to the antenna height for both the direct and reflected ray paths and is also corrected for earth curvature and average refraction. The receiver heights, z_m and z_p, relative to both the real (direct) and image (reflected) antenna height, respectively, are defined as follows:

$$z_p = zoutpa_j - \frac{r_{out}^2}{ta_{ek}}$$

where $zoutma_j$ and $zoutpa_j$ represent the output height $zout_j$ relative to the "real" and "image" antenna heights, respectively, with respect to mean sea level. ta_{ek} is twice the effective earth radius as calculated in the FILLHT SU.

2. Next, the point or range of reflection, $x_{reflect}$, is given by

$$x_{reflect} = r_{out} \frac{ant_{ref}}{z_p}$$

This quantity is used when referencing the GETREFCOEF SU.

3. The elevation angles for the direct- and reflected-path rays, α_d and α_r, respectively, are given as

$$\alpha_d = \mathrm{TAN}^{-1}\left(\frac{z_m}{r_{out}}\right)$$

$$\alpha_r = \mathrm{TAN}^{-1}\left(\frac{z_p}{r_{out}}\right)$$

4. The ANTPAT SU is referenced with the direct-path elevation angle to obtain the antenna pattern factor for the direct-path ray, $f(\alpha_d)$, and with the grazing angle (opposite of the reflected-path ray angle) to obtain the antenna pattern factor for the surface-reflected ray, $f(-\alpha_r)$.

5. The path lengths for both the direct-path, r_1, and surface-reflected path, r_2, are computed from simple right triangle calculations, as

$$r_1 = \sqrt{z_m^2 + r_{out}^2} \; ,$$

$$r_2 = \sqrt{z_p^2 + r_{out}^2} \; .$$

6. The GETREFCOEF SU is referenced with the reflected-path ray angle to obtain the amplitude, R_{mag}, and phase angle, φ, of the surface-reflection coefficient.

7. From the two path lengths, the surface-reflection phase lag angle, and the free-space wave number, k_o, the total phase angle is determined as

$$\Omega = (r_2 - r_1)\, k_o + \varphi \; .$$

8. The square of the coherent sum of both the direct-path ray and surface-reflected path ray is computed as

$$f_{sum}^2 = \left| f(\alpha_d)^2 + R_{mag}^2\ f(-\alpha_r)^2 + 2\ f(\alpha_d)\ f(-\alpha_r)\ R_{mag}^2\ \cos(\Omega) \right|.$$

9. The propagation factor in decibels, F_{fac}, is computed as

$$F_{fac} = 10\ \mathrm{LOG}\left(\mathrm{AMAX}\left(\left|f_{sum}^2\right|, 10^{-25}\right)\right).$$

   A limit of -250 dB was put on F_{fac} to avoid underflow problems.

10. Finally, the propagation loss for the output point, j, is calculated and rounded to the nearest centibel as

$$mloss_j = \mathrm{NINT}\left[10\ \left(pl_{cnst} + 20\ \mathrm{LOG}(r_1) - F_{fac}\right)\right],$$

   where pl_{cnst} is the free-space propagation loss term in decibels.

Tables 58 and 59 identify, describe, and provide units of measure and computational source for each input and output data element of the FEM SU.

Table 58. FEM SU input data element requirements.

Name	Description	Units	Source
ant_{ref}	Transmitting antenna height relative to h_{minter}	meters	TERINIT SU
ht_{lim}	Maximum height relative to h_{minter}	meters	TERINIT SU
j_{fe}	Ending index within *mloss* of FE loss values	N/A	Calling SU
j_{fs}	Starting index within *mloss* of FE loss values	N/A	Calling SU
k_o	Free-space wavenumber	meters^{-1}	APMINIT CSC
pl_{cnst}	Constant used in determining propagation loss (pl_{cnst} = 20 LOG(2 k_o))	N/A	APMINIT CSC
r_{out}	Current output range	meters	Calling SU
r_{sq}	Square of current output range	meters^2	Calling SU
ta_{ek}	Twice the effective earth's radius	meters	FILLHT SU
zoutma	Array output heights relative to "real" ant_{ref}	meters	APMINIT CSC
zoutpa	Array output heights relative to "image" ant_{ref}	meters	APMINIT CSC

Table 59. FEM SU output data element requirements.

Name	Description	Units
α_d	Direct path ray angle	radians
mloss	Propagation loss	cB
$x_{reflect}$	Range at which ray is reflected	meters

5.2.4 Free-Space Range Step (FRSTP) SU

The FRSTP SU propagates the complex PE solution in free space by one range step.

Upon entry, the PE field, *farray*, is transformed to p-space (Fourier space) and its array elements are multiplied by corresponding elements in the free-space propagator array, *frsp*. Before exiting, the PE field is transformed back to z-space. Both transforms are performed by referencing the FFT SU.

Tables 60 and 61 identify, describe, and provide units of measure and computational source for each input and output data element of the FRSTP SU.

Table 60. FRSTP SU input data element requirements.

Name	Description	Units	Source
farray	Field array to be propagated one range step in free space	µV/m	Calling SU
frsp	Complex free-space propagator term array	N/A	PHASE1 SU
n_{m1}	$n_{fft} - 1$	N/A	APMINIT CSC

Table 61. FRSTP SU output data element requirements.

Name	Description	Units
farray	Propagated field array	µV/m

5.2.5 FZLIM SU

The FZLIM SU calculates and stores the outward propagation angle and propagation factor at the top of the PE region for the current PE range. The following steps (1 through 5) are performed for each reference to the FZLIM SU.

1. The GETPFAC SU is referenced to determine the propagation factor, pf_{db}, at height, z_{lim}-y_{cur}.
2. If this is the first reference to the FZLIM SU ($iz = 1$), then the GETPFAC SU is referenced to determine the propagation factor, pf_{dblst}, at the previous PE range. A linear interpolation is performed on pf_{db} and pf_{dblst} to compute the propagation factor at range, r_{atz}, where the XO re-

gion begins. The interpolated propagation factor and the outward propagation angle, pf_{ratz} and a_{atz}, respectively, are stored in the array, *ffacz*. Next, a reference to the SAVEPRO SU is made to store the refractivity profile at the current range from height, z_{lim}, to the maximum desired output height.

3. A reference is made to the SPECEST SU to determine the outward propagation angle, ϑ_{out}. The counter, *iz*, is incremented, but is limited to iz_{max}. The propagation factor, pf_{db}, current PE range, *r*, and ϑ_{out} (with maximum limit of a_{atz}) are stored in $ffacz_{1,iz}$, $ffacz_{2,iz}$, and $ffacz_{3,iz}$, respectively.

4. If *iz* is greater than 2, then the propagation angle is checked and slightly altered to avoid extreme spiking when using these angles in the XO region. If f_{ter} is '.false.', then the angle stored in *ffacz* is the smaller of ϑ_{out} or the previously stored angle. Now, if f_{ter} is '.false.', or conversely, if f_{ter} is '.true.' and *iz* is less than or equal to 10, then the iz^{th} angle stored is adjusted and given by

$$\alpha_{dif} = ffacz_{3,iz} - ffacz_{3,iz-1}$$

$$ffacz_{3,iz} = ffacz_{3,iz-1} \pm \mathrm{AMIN}\left(\alpha_{dif}, 10^{-4}\right),$$

where '+' or '-' is used depending on the sign of α_{dif}.

5. Before exiting, a final reference to the SAVEPRO SU is made to store the refractivity profile from height, z_{lim}, to the maximum desired output height at the current range.

Tables 62 and 63 identify, describe, and provide units of measure and computational source for each input and output data element of the FZLIM SU.

Table 62. FZLIM SU input data element requirements.

Name	Description	Units	Source
a_{atz}	Local ray or propagation angle at height, z_{lim}, and range, r_{atz}	radians	APMINIT CSC
Δr_{PE}	PE range step	meters	APMINIT CSC
Δz_{PE}	PE mesh height increment (bin width in z-space)	meters	FFTPAR SU
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
iz	Number of propagation factor, range, angle triplets stored in *ffacz*	N/A	APMINIT CSC FZLIM SU

Table 62. FZLIM SU input data element requirements. (Continued)

Name	Description	Units	Source
iz_{max}	Maximum number of points allocated for arrays associated with XO calculations	N/A	APMINIT CSC
r	Current PE range	meters	Calling SU
r_{az}	Range at which z_{lim} is reached (used for hybrid model)	meters	APMINIT CSC
r_{last}	Previous PE range	meters	Calling SU
r_{log}	10 LOG(PE range, r)	N/A	PESTEP SU
r_{loglst}	10 LOG(previous PE range, r_{last})	N/A	PESTEP SU
U	Complex PE field at range, r	μV/m	PESTEP SU
$Ulast$	Complex PE field at range, r_{last}	μV/m	PESTEP SU
y_{cur}	Height of ground at current PE range, r	meters	PESTEP SU
y_{last}	Height of ground at previous range, r_{last}	meters	PESTEP SU
z_{lim}	Height limit for PE calculation region	meters	GETTHMAX SU

Table 63. FZLIM SU output data element requirements.

Name	Description	Units
$ffacz$	Array containing propagation factor, range, and propagation angle at z_{lim}	dB, meters, radians
iz	Number of propagation factor, range, angle triplets stored in ff_{acz}	N/A

5.2.6 Get Propagation Factor (GETPFAC) SU

The GETPFAC SU determines the propagation factor at a specified height.

First, linear interpolation is performed on the magnitudes of the PE field at bins k and k+1 to determine the magnitude, p_{mag}, of the field at the receiver height, z_r :

$$p_{mag} = |U_k| + f_r \left(|U_{k+1}| - |U_k| \right),$$

where the interpolation fraction, f_r, is determined from

$$f_r = \frac{z_r}{\Delta z_{PE}} - k; \quad k\Delta z_{PE} \le z_r < (k+1)\Delta z_{PE}$$

$$k = \mathrm{INT}\left(\frac{z_r}{\Delta z_{PE}}\right).$$

p_{mag} is constrained to be not less than 10^{-10} μV/m. Finally, the propagation factor, F_{fac}, is given by

$$F_{fac} = \mathrm{AMAX}\left[\,-200,\ -20\,\mathrm{LOG}\left(p_{mag}\right) - r_{log}\right].$$

Tables 64 and 65 identify, describe, and provide units of measure computational source for each input and output data element of the GETPFAC SU.

Table 64. GETPFAC SU input data element requirements.

Name	Description	Units	Source
Δz_{PE}	PE mesh height increment (bin width in z-space)	meters	Calling SU
r_{log}	10 LOG(PE range, r)	N/A	Calling SU
U	Complex PE field at range, r	μV/m	Calling SU
z_r	Receiver height	meters	Calling SU

Table 65. GETPFAC SU output data element requirements.

Name	Description	Units
F_{fac}	Propagation factor at specified height, z_r	dB

5.2.7 Get Reflection Coefficient (GETREFCOEF) SU

The GETREFCOEF SU computes the Fresnel complex reflection coefficient for a given grazing angle, ψ.

Upon entering, the proper dielectric constant, nc_i^2, to be applied to the reflected ray, must be determined. A DO WHILE loop is performed on the array, *rgrnd*, to determine the index, i, at which the range, $x_{reflect}$ (range at which ray is reflected), falls between two consecutive range points in *rgrnd*. Once this is found, the corresponding dielectric constant, nc_i^2, is used in the following equations to compute the reflection coefficient:

$$R_V = \frac{nc_i^2\,\mathrm{SIN}(\psi) - \sqrt{nc_i^2 - \mathrm{COS}^2(\psi)}}{nc_i^2\,\mathrm{SIN}(\psi) + \sqrt{nc_i^2 - \mathrm{COS}^2(\psi)}}$$

$$R_H = \frac{\mathrm{SIN}(\psi) - \sqrt{nc_i^2 - \mathrm{COS}^2(\psi)}}{\mathrm{SIN}(\psi) + \sqrt{nc_i^2 - \mathrm{COS}^2(\psi)}}$$

where R_V and R_H represent the reflection coefficients for vertical and horizontal polarization, respectively, and nc_i^2 is given by

$$nc_i^2 = \varepsilon_i + j60\sigma_i\lambda .$$

The variables, ε_i and σ_i, are the relative permittivity and conductivity, respectively, applied at range, $rgrnd_i$, and λ is the wavelength.

If the frequency is greater than 300 MHz, then for horizontal polarization, R_H, is set equal to $e^{-j\pi}$.

Tables 66 and 67 identify, describe, and provide units of measure and computational source for each input and output data element of the GETREFCOEF SU.

Table 66. GETREFCOEF SU input data element requirements.

Name	Description	Units	Source
f_{MHz}	Frequency	MHz	Calling CSCI
i_{gr}	Number of different ground types specified	N/A	Calling CSCI
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
nc^2	Array of complex dielectric constants	N/A	DIEINIT SU
ψ	Grazing angle	radians	Calling SU
$rgrnd$	Array containing ranges at which varying ground types apply.	meters	Calling CSCI
$x_{reflect}$	Range at which ray is reflected	meters	FEM SU RAYTRACE SU

Table 67. GETREFCOEF SU output data element requirements.

Name	Description	Units
$R_{V,H}$	Complex reflection coefficient for vertical (V) and horizontal (H) polarization	N/A
R_{mag}	Magnitude of the reflection coefficient: $\lvert R_{V,H} \rvert$	N/A
φ	Phase of the reflection coefficient	N/A

5.2.8 Parabolic Equation Step (PESTEP) SU

The PESTEP SU computes propagation loss at a specified range based upon the split-step Fourier PE algorithm.

Upon entering the PESTEP SU, if the current output range step, i_{stp}, is equal to 1, the current PE range, r and r_{log} (10 times the logarithm of r), are set equal to zero. An iterative DO WHILE loop then begins to advance the PE solution such that for the current PE range, a PE solution is calculated from the solution at the previous PE range. This iterative procedure is repeated in the DO WHILE loop until r is greater than the output range, r_{out}. The following steps (1 through 10) are performed for each PE range step within the DO WHILE loop.

1. First, if the current PE range, r, is greater than zero, then the height of the ground at the previous PE range, y_{last}, is set to the height of the ground at the current PE range, y_{cur}. Next, the previous PE range, r_{last}, is set equal to the current PE range r. The complex PE field, U, of the previous range is stored in the array, *Ulast*, for subsequent horizontal interpolation at range, r_{out}. This transfer of values from U to *Ulast* is made for values of the index, i, running from 0 through n_{fft}. In addition, r is incremented by one PE range step, Δr_{PE}. Finally, the range at which interpolation for range-dependent refractivity profiles is performed, r_{mid}, is also incremented by one-half the PE range step.

2. If performing a terrain case (i.e., f_{ter} is '.true.'), the ground heights, y_{cur} and y_{mid}, at range, r and r_{mid}, respectively, must be determined. If using vertical polarization, the surface impedance, α_V, must also be updated if necessary. The variables, y_{cur} and y_{mid}, are determined as

$$\begin{aligned} y_{cur} &= ty_k + slp_k\left(r - tx_k\right) \\ y_{mid} &= ty_k + slp_k\left(r_{mid} - tx_k\right)' \end{aligned}$$

   where k is the terrain profile counter, tx_k and ty_k represent the range and height, respectively, of the k^{th} point in the terrain profile, and slp_k is the slope of the k^{th} terrain segment. The current PE range is then checked against the range of the current ground type given by array *rgrnd*, and if necessary, the ground type counter, i_g, is incremented and a new α_V is computed by referencing the GETALN SU. Finally, if the current terrain slope is negative, the DOSHIFT SU is referenced to shift the field by the appropriate number of bins.

3. If using vertical polarization, regardless of the value of f_{ter}, the difference equation of the complex PE field is computed and is given by

$$w_i = \alpha_v U_i + \frac{U_{i+1} - U_{i-1}}{2\Delta z_{PE}}; \qquad i = 1,2,3,\ldots n_{fft} - 1.$$

   The array, w, is then propagated in free space one range step by referencing the FRSTP SU and the coefficients used in vertical polarization calculations, C_1 and C_2, are propagated to the new range as follows:

$$\begin{aligned} C_1 &= C_1 * C_{1x} \\ C_2 &= C_2 * C_{2x} \end{aligned}.$$

4. If using horizontal polarization, regardless of the value of f_{ter}, the PE field array, U, is propagated in free space one range step by referencing the FRSTP SU.

5. If the APM CSCI is in a range-dependent mode (i.e., the number of profiles, n_{prof}, is greater than 1), or a terrain profile is specified, the REFINTER SU is referenced to compute a new modified refractive index profile, *profint*, adjusted by the local ground height, y_{mid}, at range, r_{mid}. The PHASE2 SU is then referenced to compute a new environmental phase array, *envpr*, based on this new refractivity profile.

6. The following procedure outlines the implementation of steps 9 through 11 in Kuttler's formulation for applying the Leontovich boundary condition within the split-step PE algorithm. This procedure is only performed if using vertical polarization. First, the particular solution, *ym*, of Kuttler's difference equation is computed as follows:

$$ym_0 = 0$$
$$ym_i = 2\Delta z_{PE}\, w_i + R_T\, ym_{i-1}; \quad for\ i = 1,2,3,\ldots,n_{fft}-1,$$

   where R_T is a quadratic root as computed in the GETALN SU. The complex PE field, U, is then determined from

$$U_{n_{fft}-i} = R_T\left(ym_{n_{fft}-i} - U_{n_{fft}-i+1}\right); \quad for\ i = 1,2,3,\ldots,n_{fft}$$
$$U_{n_{fft}} = 0$$

Next, two summation terms, sum_1 and sum_2 are computed:

$$sum_1 = \frac{1}{2}\left(U_0 + U_{n_{fft}} root_{n_{fft}}\right) + \sum_{i=1}^{n_{fft}-1} U_i\, root_i$$
$$sum_2 = \frac{1}{2}\left(U_0 rootm_{n_{fft}} + U_{n_{fft}}\right) + \sum_{i=1}^{n_{fft}-1} U_{n_{fft}-i}\, rootm_i ,$$

where *root* and *rootm* are arrays of R_T and $-R_T$ to the i^{th} power, respectively. The final step in computing the field, U, for vertical polarization is

$$U_i = U_i + a\ root_i + b\ rootm_{n_{fft}-i}; \quad for\ \ i = 0,1,2,\ldots,n_{fft},$$

where

$$a = C_1 - R\ sum_1$$
$$b = C_2 - R\ sum_2,$$

$$R = \frac{2\left(1-R_T^2\right)}{\left(1+R_T^2\right)\left(1-R_T^{2n_{fft}}\right)}.$$

7. The complex field, U, is now multiplied by the environmental phase array, *envpr*, for values of the index, i, running from 0 through n_{fft}-1.

8. Next, if the current terrain slope is positive, the DOSHIFT SU is referenced to shift the field by the appropriate number of bins.

9. If XO calculations are to be performed ($i_{xo} \geq 1$) and the current PE range is greater than r_{atz}, then the FZLIM SU is referenced to determine and store the outward propagation angle at the top of the PE region for subsequent use in the EXTO SU.

10. Finally, after the output range, r_{out}, is reached and the DO WHILE loop exited, the CALCLOS SU is referenced to obtain the propagation loss values at the desired output heights at the current output range, r_{out}.

Tables 68 and 69 identify, describe, and provide units of measure and computational source for each input and output data element of the PESTEP SU.

Table 68. PESTEP SU input data element requirements.

Name	Description	Units	Source
α_V	Surface impedance term	N/A	GETALN SU
C_1	Coefficient used in vertical polarization calculations	N/A	APMINIT CSC PESTEP SU
C_2	Coefficient used in vertical polarization calculations	N/A	APMINIT CSC PESTEP SU
C_{1x}	Constant used to propagate C_1 by one range step	N/A	GETALN SU
C_{2x}	Constant used to propagate C_2 by one range step	N/A	GETALN SU
Δr_{PE}	PE range step	meters	APMINIT CSC
Δr_{PE2}	½ PE range step	meters	APMINIT CSC
Δz_{PE2}	2 Δz_{PE}	meters	APMINIT CSC
envpr	Complex [refractivity] phase term array interpolated every Δz_{PE} in height	N/A	PHASE2 SU
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
i_g	Counter indicating current ground type being modeled	N/A	APMINIT CSC PESTEP SU
i_{gr}	Number of different ground types specified	N/A	Calling CSCI

Table 68. PESTEP SU input data element requirements. (Continued)

Name	Description	Units	Source
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	N/A	Calling CSCI
i_{stp}	Current output range step index	N/A	Calling SU
i_{tpa}	Number of height/range points pairs in profile, tx, ty	N/A	APMINIT CSC
i_{xo}	Number of range steps in XO calculation region	N/A	APMINIT CSC
iz	Counter for points stored in $ffacz$	N/A	FZLIM SU
iz_{inc}	Integer increment for storing points at top of PE region (i.e., points are stored at every iz_{inc} range step)	N/A	APMINIT CSC
n_{fft}	PE transform size	N/A	FFTPAR SU
n_{m1}	$n_{fft}-1$	N/A	APMINIT CSC
n_{prof}	Number of refractivity profiles	N/A	Calling CSCI
r_{atz}	Range at which z_{lim} is reached (used for hybrid model)	meters	APMINIT CSC
$rgrnd$	Array containing ranges at which varying ground types apply	meters	Calling CSCI
R	Coefficient used in C_1 and C_2 calculations	N/A	GETALN SU
r_{log}	10 LOG(PE range, r)	N/A	PESTEP SU
r_{max}	Maximum specified range	meters	Calling CSCI
$root$	Array of R_T to the i^{th} power (e.g., $root_i = R_T^i$)	N/A	GETALN SU
$rootm$	Array of $-R_T$ to the i^{th} power (e.g., $rootm_i = (-R_T)^i$)	N/A	GETALN SU
r_{out}	Current output range	meters	Calling SU
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	N/A	GETALN SU
slp	Slope of each segment of terrain	N/A	TERINIT SU
tx	Range points of terrain profile	meters	TERINIT SU
ty	Adjusted height points of terrain profile	meters	TERINIT SU
U	Complex PE field	µV/m	PESTEP SU
y_{cur}	Height of ground at current range, r	meters	PESTEP SU

Table 69. PESTEP SU output data element requirements.

Name	Description	Units
C_1	Coefficient used in vertical polarization calculations	N/A
C_2	Coefficient used in vertical polarization calculations	N/A
i_g	Counter indicating current ground type being modeled	N/A
j_{end}	Index at which valid loss values in *mloss* end	N/A
j_{start}	Index at which valid loss values in *mloss* begin	N/A
mloss	Propagation loss	cB
r_{last}	Previous PE range	meters
r_{log}	10 LOG(PE range, r)	N/A
r_{loglst}	10 LOG(previous PE range, r_{last})	N/A
r_{mid}	Range at which interpolation for range-dependent profiles is performed	meters
U	Complex PE field at range, r	µV/m
Ulast	Complex PE field at range, r_{last}	µV/m
w	Difference equation of complex PE field	µV/m^2
y_{cur}	Height of ground at current range, r	meters
y_{curm}	Height of ground at range, $r+\Delta r_{PE2}$	meters
y_{last}	Height of ground at previous range, r_{last}	meters
ym	Particular solution of difference equation	µV/m

5.2.9 Ray Trace (RAYTRACE) SU

Using standard ray trace techniques, a ray is traced from a starting height, ant_{ref}, and range, 0, with a specified starting elevation angle, α, to a termination range, x_r. As the ray is traced, an optical path length difference, pl_d (the difference between the actual path length and x_r), and a derivative of range with respect to elevation angle, $dxd\alpha$, are being continuously computed. If the ray should reflect from the surface, a grazing angle, ψ, is determined. Upon reaching the termination range, a terminal elevation angle, β, is determined along with a termination height, z_r.

The ray trace is conducted by stepping in profile levels and computing ending values. A number of stepping scenarios, based upon starting and ending elevation angles, determine the program flow of the RAYTRACE SU. These scenarios are a ray that is upgoing, a ray that is downgoing, and a ray that turns around within a layer.

Upon entering the SU, a running range, x_{sum}, the range at which a ray is reflected, $x_{reflect}$, $dxd\alpha$, pl_d, ψ, and a ray type (direct or reflected) flag, i_{type}, are initialized to zero. A temporary beginning eleva-

tion angle, a_{start}, is set equal to α, and an environmental profile level counter, i, is set equal to the array index for the height in the RO region corresponding to the transmitter height, i_{start}.

The following steps (1 and 2) are now repeated while x_{sum} remains less than the termination range, x_r. Upon failure to meet this repetition criterion, the SU program flow continues with step 3 below.

1. The beginning angle, a_{start}, is examined to determine if the ray is initially upgoing (i.e., $a_{start} \geq 0$) or downgoing. If it is upgoing, the SU program flow continues with steps 1a through 1e; otherwise the program flow continues with step 2 below.

   a. The level counter is examined and if the ray is in the highest layer (i.e., $i = l_{evels}$), the ending angle, height, range/angle derivative, and path length difference are given as

$$\beta = a_{start} + (x_r - x_{sum})\, gr_i,$$

$$z_r = zrt_i + \frac{\beta^2 - a_{start}^2}{2\, gr_i},$$

$$dxd\alpha = dxd\alpha + \frac{\left[\frac{\alpha}{\beta} - \frac{\alpha}{a_{start}}\right]}{gr_i},$$

$$pl_d = pl_d + \frac{\left(rm_i - \frac{a_{start}^2}{2}\right)(\beta - a_{start}) + \frac{\beta^3 - a_{start}^3}{3}}{gr_i},$$

   respectively, where *gr* is an intermediate M-unit gradient, *rm* is an intermediate M-unit, and *zrt* is an intermediate height. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 3 below.

   b. If the ray is not in the highest layer, the ray must be examined to determine if it will turn around and become a downgoing ray within the current layer. This is done by looking at the radical term, *rad*, which will be used in the ending angle calculation. The radical term is given as

$$rad = a_{start}^2 + q_i,$$

   where q is an intermediate M-unit difference. If *rad* is greater than or equal to zero, a solution for the ending angle is possible. The ray will not turn around and the program flow continues with step c; otherwise the program flow continues with step 1d.

   c. Before calculations can continue, the possible ending range must be compared to the termination range. This possible ending range is determined as

$$x_{temp} = x_{sum} + \frac{\beta - a_{start}}{gr_i},$$

$$\beta = \sqrt{rad}\,.$$

This possible ending range is compared to the termination range and if it is larger, the ending angle, height, range/angle derivative, and path length difference are computed from equations given in step 1a. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 3 below.

If the ray has not reached the termination range, x_{sum} is updated to x_{temp} ; the range/angle derivative and path length difference are computed from equations given in step 1a, where $\beta = \sqrt{rad}$; a_{start} is updated to β; the level counter is incremented by one; and the program flow returns to the top of step 1 above.

d. If the ray has, in fact, turned around within the current layer, a determination must be made for the ray reaching a full range step within the still upgoing segment, for the ray reaching a full range step within the downgoing segment, or the ray exceeding the termination range. The full range step is given by

$$x_{temp} = x_{sum} - \frac{a_{start}}{gr_i},$$

which is compared to the termination range. If it exceeds the termination range, the ending angle, the ending height, the range/angle derivative, and the path length difference are determined from equations given in step 1a. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 3 below. If the termination range has not been exceeded, further examination of the ray's segments must be made.

e. At this point, x_{sum} is updated to x_{temp}; x_{temp} is recalculated as shown in step 1d; and x_{temp} is again compared to the termination range. If the termination range has been exceeded, the ending angle is given as

$$\beta = (x_r - x_{sum})\, gr_i,$$

and the ending height, the range/angle derivative, and the path length difference are now determined from equations shown in step 1a. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 3 below.

If the termination range has not been exceeded, x_{sum} is updated to x_{temp}; β is updated to a_{start}; the range/angle derivative and path length difference are determined from equations shown in step 1a; a_{start} is updated to β; and the program flow returns to the top of step 1 above.

2. **Note!** The equations for the upgoing ray within step 1 above apply equally to the downgoing ray except where specified otherwise. However, in applying these equations to step 2, the level counter, *i*, within the intermediate M-unit gradient sub-term, *gr*, must be reduced by one.

The beginning angle, a_{start}, has been examined in step 1 above and the ray has been determined to be initially downgoing. Similar to step 1 above, the ray must be examined to determine if it has turned around and has become an upgoing ray within the current layer. This

is done by looking at the radical term, *rad*, which will be used in the ending angle calculation. This radical term is given as

$$rad = a_{start}^2 - q_{i-1}.$$

If *rad* is greater than or equal to zero, a solution for the ending angle is possible. The ray has not turned around and the program flow continues with steps 2a through 2c below; otherwise the program flow continues with step 2d.

a. Before calculations can continue, the possible ending range must be compared to the termination range. This possible ending range is determined from the equation given for x_{temp} in step 1c, where β is now $-\sqrt{rad}$. This possible ending range is compared to the termination range and if it is larger, the ending angle, the ending height, the range/angle derivative, and the path length difference are computed from equations shown in step 1a. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 2c below.

b. If the termination range has not been exceeded, x_{sum} is updated to x_{temp}; the range/angle derivative and path length difference are computed as shown in step 1a, where $\beta = -\sqrt{rad}$; a_{start} is updated to β; and the level counter is decremented by one.

c. The level counter is examined, and if it is zero, the ray has reflected from the surface. In this case, the ray type flag is set to 1 to indicate a reflection; the grazing angle is set as $\psi = |a_{start}|$, and $x_{reflect}$ is set equal to x_{temp}. At this point a symmetry check is made. The idea of symmetry says that the ray will return to its starting height, at twice the reflection range, with an ending elevation angle opposite the starting elevation angle. Symmetry is used for APM speed efficiency so as to preclude redundant ray trace calculations on the upward path back to the starting height. Prior to applying symmetry, however, the possible ending range (twice x_{sum}) must be compared to the termination range. If the termination range is exceeded by making the symmetry assumption, a_{start} is updated to $-a_{start}$ and the assumption is vacated. If not, however, the assumption is invoked and a_{start} is updated to $-\alpha$; x_{sum} , $dxd\alpha$, and pl_d, are doubled; and the level counter is restored to i_{start}. Control is now returned to the top of step 1 above.

d. From step 2, the ray has turned around within the current layer and is now an upgoing ray. Similar to the upgoing case of step 1, a determination must be made for the ray reaching a full range step within the still downgoing segment, for the ray reaching a full range step within the upgoing segment, or the ray exceeding the termination range. The full range step is given by x_{temp} as computed in step 1d.

   If the full range step exceeds the termination range, the ending angle, the ending height, the range/angle derivative, and the path length difference are computed from equations shown in step1a. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 3 below. If the termination range has not been exceeded, further examination of the ray's segments must be made.

ending angle is determined as in step 1e; the ending height, range/angle derivative, and path length difference are determined as in step 1a. Satisfaction of this condition causes the failure of the repetition criterion and the SU program flow continues with step 3 below.

If the termination range has not been exceeded, x_{sum} is updated to x_{temp}; β is updated to $-a_{start}$; the range/angle derivative and path length difference are determined as in step1a; a_{start} is updated to β; and the program flow returns to the top of step 1 above.

3. Within APM, the terminal elevation angle is not allowed to equal zero. Therefore, if its absolute value is less than 10^{-10}, it is reset to 10^{-10} while retaining its present sign.

Tables 70 and 71 identify, describe, and provide units of measure and computational source for each input and output data element of the RAYTRACE SU.

Table 70. RAYTRACE SU input data element requirements.

Name	Description	Units	Source
α	Source elevation angle	radians	Calling SU
gr	Intermediate M-unit gradient array, RO region	(M-unit/m)10^{-6}	REFINIT SU
i_{start}	Array index for height in RO region corresponding to ant_{ref}	N/A	REFINIT SU
$levels$	Number of levels in gr, q, and zrt arrays	N/A	REFINIT SU
q	Intermediate M-unit difference array, RO region	2(M-unit)10^{-6}	REFINIT SU
rm	Intermediate M-unit array, RO region	M-unit 10^{-6}	REFINIT SU
x_r	Terminal range—called x_{ROn} in ROCALC SU	meters	Calling SU
zrt	Intermediate height array, RO region	meters	REFINIT SU

Table 71. RAYTRACE SU output data element requirements.

Name	Description	Units
β	Terminal elevation angle	radians
$dxd\alpha$	Derivative of range with respect to elevation angle	meters/radians
i_{type}	Ray type (direct or reflected) flag	N/A
pl_d	Path length from range, x_r	meters
ψ	Grazing angle	radians
$x_{reflect}$	Range at which ray is reflected	meters
z_r	Terminal height	meters

5.2.10 Refractivity Interpolation (REFINTER) SU

The REFINTER SU interpolates both horizontally and vertically on the modified refractivity profiles. Profiles are then adjusted so they are relative to the local ground height.

First, an in-line function for linear interpolation is defined by

$$\mathrm{PINT}\,(\,p_1,\,p_2\,)\;=\;p_1\;+\;fv\;(\,p_2\;-\;p_1\,)\,.$$

Upon entry, the number of height/refractivity levels , *lvlep*, for the current profile is set equal to the user-specified number of levels for all profiles specified, *lvlp*. For the range-dependent case, all profiles have the same number of levels.

If there is a range-dependent environment (i.e. ,$n_{prof} > 1$), horizontal interpolation to r_{ange} is performed between the two neighboring profiles that are specified relative to mean sea level. In this case, the following calculations are made. If r_{ange} is greater than the range for the next refractivity profile rv_2, then the index, j (indicating the range of the previous refractivity profile), is set equal to the counter for the range of the current profile, i_s; i_s is then incremented by one. Next, the range of the previous refractivity profile, rv_1, is set equal to rv_2, and rv_2 is set equal to the range of the i_s^{th} profile, $rngprof_{i_s}$. The fractional range fv for the interpolation is given by

$$fv\;=\;\frac{r_{ange}\;-\;rv_1}{rv_2\;-\;rv_1}\;.$$

The array, *refdum,* containing M-unit values for the current (interpolated) profile and the array, *htdum*, containing height values for the current (interpolated) profile, are determined from

$$refdum_i\;=\;\mathrm{PINT}\left[refmsl_{i,j}, refmsl_{i,i_s}\right];\quad i=1,2,3,\ldots lvlep$$

$$htdum_i\;=\;\mathrm{PINT}\left[hmsl_{i,j}, hmsl_{i,i_s}\right];\quad i=1,2,3,\ldots lvlep$$

where *refmsl* and *hmsl* are two-dimensional arrays containing refractivity and height, respectively, with respect to mean sea level of each user-specified profile.

The REMDUP SU is referenced to remove duplicate refractivity levels, with *lvlep* being the number of points in the profile at range, r_{ange}. The PROFREF SU is then referenced to adjust the new profile (i.e., *refdum* and *htdum*) relative to the internal reference height, h_{minter}, corresponding to the minimum height of the terrain profile. The PROFREF SU is then referenced once more to adjust the profile relative to the local ground height, y_{curm}, and upon exit from the PROFREF SU, the INTPROF SU is referenced to interpolate vertically on the refractivity profile at each PE mesh height point. This results in the n_{fft}-point profile array, *profint*, which contains the interpolated M-unit values for the refractivity at r_{ange}, where n_{fft} is the transform size.

Upon exiting the REFINTER SU, rv_1 and the index, j, are saved for use upon the next reference of the SU.

Tables 72 and 73 identify, describe, and provide the units of measure and computational source for each input and output data element of the REFINTER SU.

Table 72. REFINTER SU input data element requirement.

Name	Description	Units	Source
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
h_{minter}	Minimum height of terrain profile	meters	TERINIT SU
hmsl	Two-dimensional array containing heights with respect to mean sea level of each profile. Array format must be $hmsl_{i,j}$ = height of i^{th} level of j^{th} profile. j = 1 for range-independent cases	meters	Calling CSCI
i_s	Counter for current profile	N/A	REFINIT SU REFINTER SU
i_{stp}	Current output range step index	N/A	Calling SU
lvlp	Number of height/refractivity levels in profiles	N/A	Calling CSCI
n_{prof}	Number of refractivity profiles	N/A	Calling CSCI
r_{ange}	Range for profile interpolation	meters	Calling SU
refmsl	Two-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be $refmsl_{i,j}$ = M-unit at i^{th} level of j^{th} profile. j = 1 for range-independent cases	M-unit	Calling CSCI
rngprof	Ranges of each profile. $rngprof_i$ = range of i^{th} profile	meters	Calling CSCI

Table 72. REFINTER SU input data element requirement. (Continued)

Name	Description	Units	Source
rv_2	Range of the next refractivity profile	meters	REFINIT SU REFINTER SU
y_{curm}	Height of ground midway between last and current PE range	meters	PESTEP SU

Table 73. REFINTER SU output data element requirements.

Name	Description	Units
htdum	Height array for current interpolated profile	meters
i_s	Counter for current profile	N/A
lvlep	Number of height/refractivity levels in profile *refdum* and *htdum*	N/A
profint	Profile interpolated to every Δz_{PE} in height	M-units
refdum	M-unit array for current interpolated profile	M-units
rv_2	Range of the next refractivity profile	meters

5.2.11 Remove Duplicate Refractivity Levels (REMDUP) SU

The REMDUP SU removes any duplicate refractivity levels in the current interpolated profile. Adjoining profile levels are checked to see if the heights are within 0.001 meters. If they are, the duplicate level in the profile is removed. This process continues until all profile levels (*lvlep*) have been checked.

Tables 74 and 75 identify, describe, and provide the units of measure and computational source for each input and output data element of the REMDUP SU.

Table 74. REMDUP SU input data element requirements.

Name	Description	Units	Source
htdum	Height array for current interpolated profile	meters	REFINTER SU
lvlep	Number of height/refractivity levels in profile	N/A	REFINTER SU
refdum	M-unit array for current interpolated profile	M-units	REFINTER SU

Table 75. REMDUP SU output data element requirements.

Name	Description	Units
htdum	Height array for current interpolated profile	meters
lvlep	Number of height/refractivity levels in profile	N/A
refdum	M-unit array for current interpolated profile	M-units

5.2.12 Ray Optics Calculation (ROCALC) SU

The ROCALC SU computes the RO components that will be needed (by the ROLOSS SU) in the calculation of propagation loss at a specified range and height within the RO region. These components are the magnitudes for a direct-path and surface-reflected ray, Fd^2 and Fr^2, respectively, and the total phase lag angle, Ω, between the direct-path and surface-reflected rays.

The RO region may be visualized as having a grid of points superimposed upon it. The grid points are defined at the intersection of a series of lines sloping upward from the origin and a series of vertical lines at varying ranges. The grid point counter, k, and the vertical lines are defined at varying ranges, two of which are represented by the terms x_{ROp}, a range for which the RO calculations were previously performed, and, x_{ROn}, the next calculation range.

The sloping line with the greatest angle (indicated by $k = k_{max}$) is a function of the maximum APM output height, ht_{ydif}, adjusted for terrain and reference heights, and the next calculation range. The sloping line with the least angle (indicated by $k = k_{minp}$) is a function of the height at the top of the PE region and the range of the previous RO calculations.

The following steps (1 through 4) are performed while the current range, x, is greater than x_{ROn}.

1. The terms of table 76 (defined in table 77) are initialized or updated based upon the RO calculation range counter, i_{ROp}. If i_{ROp} equals –1; the terms are initialized; otherwise the terms are updated. It should be noted that the terms must be computed in the order they appear in the table to ensure proper values are assigned to component terms.

Table 76. RO region indices, angles, and ranges

For $iROp$ = -1 (initialize terms)	For $iROp \neq$ -1 (update terms)
$i_{ROp} = 1$	$i_{ROp} = 1 - i_{ROp}$
$i_{ROn} = 0$	$i_{ROn} = 1 - i_{ROn}$
N/A	$x_{ROp} = x_{ROn}$
$k_{minp} = 0$	$k_{minp} = k_{minn}$
$k_{minn} = 0$	$k_{minn} = 0$
$ht_{ydif} = ht_{lim} - y_{fref}$	N/A
$d\alpha = \text{AMIN}\left(\frac{\theta_{bwr}}{2}, .01745\right)$	N/A
$k_{max} = 88$	$k_{max} = \text{AMIN}\left[88, \text{INT}\left(\frac{1000\, ht_{ydif}}{x_{ROp}}\right) + 2\right]$
$frac_{RO} = 0$	$frac_{RO} = \frac{1}{\left(\text{AMAX}\left[\frac{0.001\, k_{max}}{d\alpha}, 5\right] - 1\right)}$ $for\ \ frac_{RO} < 0.25$
N/A	$\Delta x_{RO} = frac_{RO}\ x_{ROp}$
$x_{ROn} = x$	$x_{ROn} = x_{ROp} + \Delta x_{RO}$

2. To calculate the RO components at each vertical point for the next range, x_{ROn}, a ray trace within a Newton iteration method is used to find a direct-path ray and a surface-reflected ray that will both originate at the transmitter height, ant_{ref}, and terminate at the same grid point, z_k. The results of the iteration are examined and if either of the rays has not been found, an adjustment in the lower boundary of the RO region is made. Following the conclusion of the iterations, the antenna pattern factors for each ray are obtained, a surface reflection coefficient for the surface-reflected ray is computed, and the RO components are calculated.

   Prior to all calculations for each vertical point, the ray trace must be initialized with beginning direct-path and surface-reflected ray elevation angles, α_d and α_r, respectively, and derivatives of height with respect to these elevation angles, $dzd\alpha_d$ and $dzd\alpha_r$. A starting assumption is made that the direct-path ray and the surface-reflected rays are parallel to each other. Thus, α_d is initialized as 0.001 k_{max} and α_r is initialized as $-\alpha_d$. The RAYTRACE SU is referenced separately with α_d and α_r to obtain termination elevation angles, β_d and β_r, and the two derivatives of range with respect to elevation angle, $dxd\alpha_d$ and $dxd\alpha_r$, which are

used, in turn, to compute the needed derivatives of height with respect to elevation angle given as $-\beta_d\, dxd\alpha_d$ and $-\beta_r\, dxd\alpha_r$.

3. Once the raytrace has been initialized, the following steps (a through f) are performed for each vertical grid point, z_k, beginning with $k = k_{max}$ and subsequently decrementing k downward while k remains $\geq k_{minn}$. Once k has reached zero, processing continues with step 4 below.

   a. The termination height is computed as

   $$z_k = x_{ROn}\ 0.001\, k,$$

   where k is the grid point counter.

   b. The Newton iteration method to find the direct path ray from ant_{ref} to z_k is started. This iteration is continued until the difference between the ray trace ending height z_d and z_k is less than a height difference tolerance, z_{tol}, but in any case, no more than 10 times. The direct-path elevation angle is given as

   $$\alpha_d = \alpha_d - \frac{z_d - z_k}{dzd\alpha_d},$$

   where z_d and $dzd\alpha_d$ are obtained from the ray trace initialization of step 2 above for the first iteration and from the previous iteration for subsequent iterations.

   The RAYTRACE SU is referenced and a new $dzd\alpha_d$ is calculated as $-\beta_d\, dxd\alpha_d$. This new $dzd\alpha_d$ is examined and if it is less than 10^{-6}, or if the ray type flag, i_{type}, returned from the RAYTRACE SU, indicates the ray has reflected, the lower boundary of the RO region is adjusted by setting k_{minn} equal to one more than k and the iteration for the direct ray is stopped.

   c. The Newton iteration method to find the surface-reflected ray from ant_{ref} to z_k is now started. This iteration should be continued until the difference between the ray trace ending height, z_r and z_k is less than a height difference tolerance, z_{tol}, but in any case, no more than 10 times. The reflected-path elevation angle is given as

   $$\alpha_r = \alpha_r - \frac{z_r - z_k}{dzd\alpha_r},$$

   where z_r and $dzd\alpha_r$ are obtained from the ray trace initialization of step 2 above for the first iteration and from the previous iteration for subsequent iterations.

   The RAYTRACE SU is referenced and a new $dzd\alpha_r$ is calculated as $-\beta_r\, dxd\alpha_r$. This new $dzd\alpha_r$ is examined and if it is less than 10^{-6}, or if i_{type} indicates the ray is a direct ray, the lower boundary of the RO region is adjusted by setting k_{minn} equal to one more than k and the iteration for the surface-reflected ray is stopped.

d. A test is made to determine if the grazing angle, ψ (returned from the RAYTRACE SU) is less than the limiting value, ψ_{lim}, and, if so, the lower boundary of the RO region is adjusted by setting k_{minn} equal to k.

e. The magnitudes for the direct-path and surface-reflected ray, Fd^2 and Fr^2, respectively, are now given as

$$Fd^2 = \left|\frac{x_{ROn}}{dzd\alpha_d}\right| f^2(\alpha_d) ,$$

$$Fr^2 = \left|\frac{x_{ROn}}{dzd\alpha_r}\right| \left[f(\alpha_r)\, R_{mag}\right]^2 ,$$

where the amplitude of the surface reflection coefficient, R_{mag}, is obtained from a reference to the GETREFCOEF SU; the antenna pattern factors, $f(\alpha_d)$ and $f(\alpha_r)$, are obtained from references to the ANTPAT SU, and the derivatives of height with respect to elevation angle are obtained from the RAYTRACE SU within the Newton iteration of steps 3b and 3c above.

f. The total phase lag between the direct-path and surface-reflected rays is computed as

$$\Omega = \left(pl_r - pl_d\right) k_o + \varphi ,$$

where the ray path lengths pl_d and pl_r are obtained from the RAYTRACE SU within the Newton iteration of steps 2 and 3 above; the reflection coefficient phase lag angle, φ, is obtained from a reference to the GETREFCOEF SU; and k_o is the free-space wave number.

4. If the point counter, k, has been reduced to zero by the procedures of steps 3a through 3f above, the surface values of magnitudes for the direct-path and surface-reflected rays are both set equal to the last value of Fd^2 and the total phase lag between the direct-path and surface-reflected rays is set equal to π.

Tables 77 and 78 identify, describe, and provide units of measure and computational source for each input and output data element of the ROCALC SU.

Table 77. ROCALC SU input data element requirements.

Name	Description	Units	Source
μ_{bwr}	Antenna vertical beamwidth	radians	Calling CSCI
$d\alpha$	μ_{bwr} / 2	radians	ROCALC SU
$frac_{RO}$	RO range interval fraction (0.0 to 0.25)	N/A	ROCALC SU
ht_{lim}	Maximum height relative to h_{minter}	meters	TERINIT SU
ht_{ydif}	ht_{lim} - y_{fref}	meters	ROCALC SU

Table 77. ROCALC SU input data element requirements. (Continued)

Name	Description	Units	Source
i_{ROn}	Array index for next range in RO region	N/A	ROCALC SU
i_{ROp}	Array index for previous range in RO region	N/A	APMINIT CSC ROCALC SU
k_o	Free-space wave number	meters^{-1}	APMINIT CSC
k_{minn}	Array index for minimum angle in RO region at range, x_{ROn}	N/A	ROCALC SU
ψ_{lim}	Grazing angle of limiting ray	radians	APMINIT CSC
x	Current range	meters	Calling SU
x_{ROn}	Next range in RO region	meters	APMINIT CSC
x_{ROp}	Previous range in RO region	meters	ROCALC SU
y_{fref}	Ground elevation height at source	meters	APMINIT CSC
z_{tol}	Height tolerance for Newton's method	meters	APMINIT CSC

Table 78. ROCALC SU output data element requirements.

Name	Description	Units
$d\alpha$	$\theta_{bwr}/2$	radians
Δx_{RO}	RO range interval	meters
Fd^2	Magnitude array, direct ray	N/A
Fr^2	Magnitude array, reflected ray	N/A
$frac_{RO}$	RO range interval fraction (0.0 to 0.25)	N/A
ht_{ydif}	$ht_{lim} - y_{fref}$	meters
i_{ROn}	Array index for next range in RO region	N/A
i_{ROp}	Array index for previous range in RO region	N/A
k_{max}	Array index for maximum angle in RO region at range, x_{ROn}	N/A
k_{minn}	Array index for minimum angle in RO region at range, x_{ROn}	N/A
k_{minp}	Array index for minimum angle in RO region at range, x_{ROp}	N/A
Ω	Total phase angle array	radian
x_{ROn}	Next range in RO region	meters
x_{ROp}	Previous range in RO region	meters

5.2.13 Ray Optics Loss (ROLOSS) SU

The ROLOSS SU calculates propagation loss values at a specified range and height based upon the components of magnitude for a direct-path and surface-reflected ray, Fd^2 and Fr^2, respectively, and the total phase lag angle, Ω, between the two rays as determined by the ROCALC SU.

For purposes of computational efficiency, an interpolation from the magnitude and total phase lag angle arrays established by the ROCALC SU is made to obtain these three quantities at each APM vertical output mesh point within the RO region.

From the interpolated phase lag angle and ray magnitudes a propagation factor is calculated that is used, in turn, with the free-space propagation loss to obtain a propagation loss at each vertical APM output point.

Upon entering the SU, a range ratio term to be used within the interpolation scheme is defined as

$$ratio = \frac{r_{out} - x_{ROp}}{\Delta x_{RO}}.$$

The phase lag angle and ray magnitude arrays have been filled at grid points defined by a series of sloping lines and the next and previous RO calculation range, x_{ROn} and x_{ROp}, respectively. Which values to interpolate from are determined by the sloping line immediately above and the sloping line immediately below the current APM output point of interest. To begin the calculations, k_{lo} is initialized to k_{max}, the line with the greatest angle.

The following steps (1 through 4) are now taken, decrementing downward in APM output points from the maximum output height index in the RO region, j_{max}, to the minimum output height index in the RO region, j_{min}.

1. Interpolation of Fd^2, Fr^2, and Ω values occurs in two stages. The first stage is horizontally, above and below the APM output point (i.e., along the lines k_{lo} and k_{hi}). These values will be used in turn, in a vertical interpolation stage to obtain values at the APM output point itself. It may be, however, that more than one APM output point will fall between two adjacent k lines. In this case, it would be redundant to perform the horizontal interpolation more than once. For this reason, a temporary k line counter is established that will be used in comparison with k_{lo} to determine if interpolation is necessary or if the previously interpolated horizontal values may again be used in the vertical interpolation. This temporary k counter is given by

$$k_{temp} = \mathrm{INT}\left(\frac{1000\ zRO_j}{r_{out}}\right),$$

where j is the APM output point counter and zRO_j is the j^{th} output height point . If k_{temp} is less than the current k_{lo}, the APM output point occurs below the current lower k line and horizontal interpolations must be performed using the following steps (a through c); otherwise the horizontal interpolations are unnecessary and the SU may proceed with step 2.

a. The lower k line, k_{lo}, is reset to k_{temp} and the upper k line, k_{hi}, is set to one more than k_{lo}.

b. In preparation for the interpolation, component terms (horizontal differences of direct and surface-reflected magnitudes and phase lag angles) along the k_{lo} and k_{hi} lines are given as

$$\Delta Fd_{lo}^2 = Fd_{i_{ROn},k_{lo}}^2 - Fd_{i_{ROp},k_{lo}}^2,$$

$$\Delta Fr_{lo}^2 = Fr_{i_{ROn},k_{lo}}^2 - Fr_{i_{ROp},k_{lo}}^2,$$

$$\Delta\Omega_{lo} = \Omega_{i_{ROn},k_{lo}} - \Omega_{i_{ROp},k_{lo}},$$

substituting the index k_{hi} for k_{lo} as appropriate. Note that these horizontal differences need only be calculated while both k_{hi} and k_{lo} remain greater than or equal to both k_{minp} and k_{minn}. If these conditions are not met, any continued difference calculations would take place within the PE region, which would yield undesirable results. For failure of these conditions, the previously calculated difference values are used for the lower RO region boundary calculations.

c. If k_{lo} is greater than or equal to k_{minp}, the horizontally interpolated direct and surface-reflected magnitudes and phase lag angles along the k_{lo} line can proceed in a forward manor (from x_{ROp} to r_{out}). These values are given as

$$Fd_{lo}^2 = Fd_{i_{ROp},k_{lo}}^2 + ratio\ \Delta Fd_{lo}^2,$$

$$Fr_{lo}^2 = Fr_{i_{ROp},k_{lo}}^2 + ratio\ \Delta Fr_{lo}^2,$$

$$\Omega_{lo} = \Omega_{i_{ROp},k_{lo}} + ratio\ \Delta\Omega_{lo}.$$

In a like manor, the same three equations above are used to get the values along the k_{hi} line by substituting the index, k_{hi}, assuming, however, k_{hi} is also greater than or equal to k_{minp}. Should either k_{lo} or k_{hi} be less than k_{minp}, the interpolation must proceed in a backward manor (from x_{ROn} to r_{out}). The above three equations may again be used by substituting the index, i_{ROn} for i_{ROp}, and the value (1 - *ratio*) for *ratio.*

2. Once the horizontal interpolation of magnitudes and phase lag angles has been accomplished, the vertical interpolation of magnitudes and phase lag angles at the APM output point may proceed as

$$Fd^2 = Fd_{lo}^2 + ratio_k \left(Fd_{hi}^2 - Fd_{lo}^2\right),$$

$$Fr^2 = Fr_{lo}^2 + ratio_k \left(Fr_{hi}^2 - Fr_{lo}^2\right),$$

$$\Omega = \Omega_{lo} + ratio_k \left(\Omega_{hi} - \Omega_{lo}\right),$$

where ratio from k_{lo} to k_{hi} is

$$ratio_k = \frac{1000\ zRO_j}{r_{out}} - k_{lo}.$$

3. From the magnitudes of the direct and surface-reflected components and the phase lag angle, the square of the propagation factor at the APM output point is given as

$$F^2 = \left| Fd^2 + Fr^2 + 2\sqrt{\left|Fd^2\ Fr^2\right|}\ \cos(\Omega) \right|,$$

which, in turn, is converted to a propagation factor expressed in decibels by

$$F_{fac} = 10\ \mathrm{LOG}\left[\mathrm{AMAX}\left(F^2, 10^{-25}\right)\right].$$

4. Finally, the propagation loss at the APM output point is calculated to the closest integer in centibels as

$$mloss_j = \mathrm{NINT}\left[10\ \mathrm{LOG}\left(fslr_{i_{stp}} - F_{fac}\right)\right],$$

where $fslr_{i_{stp}}$ is the free space loss at the $i_{stp}^{\ th}$ output range.

Tables 79 and 80 identify, describe, and provide units of measure and computational source for each input and output data element of the ROLOSS SU. Table 81 identifies terms that are used internal to the ROLOSS SU and whose value must be retained from SU call to SU call for reasons of computational efficiency.

Table 79. ROLOSS SU input data element requirements.

Name	Description	Units	Source
Δx_{RO}	RO range interval	meters	ROCALC SU
Fd^2	Magnitude array, direct ray	N/A	ROCALC SU
Fr^2	Magnitude array, reflected ray	N/A	ROCALC SU
$fslr$	Free-space loss array for output ranges	dB	APMINIT CSC
i_{ROn}	Array index for next range in RO region	N/A	ROCALC SU
i_{ROp}	Array index for previous range in RO region	N/A	ROCALC SU
i_{stp}	Current output range step index	N/A	Calling SU
j_{max}	Array index for maximum output height in RO region	N/A	Calling SU
j_{min}	Array index for minimum output height in RO region	N/A	Calling SU

Table 79. ROLOSS SU input data element requirements. (Continued)

Name	Description	Units	Source
K_{max}	Array index for maximum angle in RO region at range x_{ROn}	N/A	ROCALC SU
K_{minn}	Array index for minimum angle in RO region at range x_{ROn}	N/A	ROCALC SU
K_{minp}	Array index for minimum angle in RO region at range x_{ROp}	N/A	ROCALC SU
Ω	Total phase angle array	radians	ROCALC SU
r_{out}	Current output range	meters	Calling SU
x_{ROp}	Previous range in RO region	meters	ROCALC SU
zRO	Array of output heights in RO region	meters	APMINIT CSC

Table 80. ROLOSS SU output data element requirements.

Name	Description	Units
$mloss$	Propagation loss	cB

Table 81. ROLOSS SU save data element requirements.

Name	Description	Units	Source
$\Delta\Omega_{hi}$	Difference in total phase lag angle along Δx_{RO} above desired APM output point	radians	ROLOSS SU
$\Delta\Omega_{lo}$	Difference in total phase lag angle along Δx_{RO} below desired APM output point	radians	ROLOSS SU
ΔFd_{lo}^2	Difference in direct ray magnitude along Δx_{RO} below desired APM output point	N/A	ROLOSS SU
ΔFd_{hi}^2	Difference in direct ray magnitude along Δx_{RO} above desired APM output point	N/A	ROLOSS SU
ΔFr_{lo}^2	Difference in reflected ray magnitude along Δx_{RO} below desired APM output point	N/A	ROLOSS SU
ΔFr_{hi}^2	Difference in reflected ray magnitude along Δx_{RO} above desired APM output point	N/A	ROLOSS SU

5.2.14 Ray Optics Model (ROM) SU

The ROM SU serves as a one-call routine for the RO model. It performs RO calculations by referencing the ROCALC SU and determines the loss at specified height output points by referencing the ROLOSS SU.

Tables 82 and 83 identify, describe, and provide units of measure and computational source for each input and output data element of the ROM SU.

Table 82. ROM SU input data element requirements.

Name	Description	Units	Source
i_{stp}	Current output range step index	N/A	Calling SU
jre	Ending index within *mloss* of RO loss values	N/A	Calling SU
jrs	Starting index within *mloss* of RO loss values	N/A	Calling SU
r_{out}	Current output range	meters	Calling SU

Table 83. ROM SU output data element requirements.

Name	Description	Units
mloss	Propagation loss	cB

5.2.15 Save Profile (SAVEPRO) SU

The SAVEPRO SU stores the gradients and heights of the current refractivity profile upon each reference to the FZLIM SU from the top of the PE calculation region to the maximum user-specified height.

Upon entering, the current profile height array, *htdum* is searched to find the index, *i*, such that $htdum_i$ is the first height in the profile that is greater than the maximum PE calculation height, z_{lim}. The counter, l_{new}, is then initialized to -1.

Next, the gradients are calculated and stored, along with corresponding heights, as follows

$$grad_{l_{new},iz} = \frac{refdum_{j+1} - refdum_j}{htdum_{j+1} - htdum_j},$$

$$htr_{l_{new},iz} = htdum_j,$$

where j is incremented by one from i to $lvlep$-1, l_{new} is incremented by one with each increment in j, and iz represents the range step index for XO calculations.

Before exiting, the last height level in *htdum* is stored and the final number of levels, l_{new}, in the iz^{th} profile (represented by *grad* and *htr*) is stored in the array, *lvl*.

Tables 84 and 85 identify, describe, and provide units of measure and computational source for each input and output data element of the SAVEPRO SU.

Table 84. SAVEPRO SU input data element requirements.

Name	Description	Units	Source
htdum	Height array for current profile	meters	REFINTER SU
iz	Number of calculation range steps for XO region	N/A	FZLIM SU
lvlep	Number of height/refractivity levels in profile *refdum* and *htdum*	N/A	REFINTER SU
refdum	M-unit array for current profile	M-units	REFINTER SU
z_{lim}	Maximum height in PE calculation region	meters	FFTPAR SU

Table 85. SAVEPRO output data element requirements.

Name	Description	Units
grad	Two-dimensional array containing gradients of each profile used in XO calculations	M-units /meter
htr	Two-dimensional array containing heights of each profile used in XO calculations	meters
lvl	Number of height levels in each profile used in XO calculations	N/A

5.2.16 Spectral Estimation (SPECEST) SU

The SPECEST SU determines the outward propagation angle at the top of the PE calculation region based on spectral estimation.

Upon entering the SPECEST SU, the topmost n_p points (within the unfiltered portion) of the complex PE field are separated into their real and imaginary components, *xp* and *yp*, respectively. A window filter is then applied to both arrays by multiplying each element in *xp* and *yp* by each corresponding element in the filter array, *filtp*, for indices between ¾ n_p and n_p.

Next, the array elements in *xp* and *yp* are set to 0 for indices from n_p+1 to n_s-1. (Note that both *xp* and *yp* are arrays of size n_s.) The SINFFT SU is then referenced to obtain the spectral field components.

The spectral amplitudes in dB are then given by

$$spectr_i = 10\ \mathrm{LOG}\left[\mathrm{AMAX}\left(10^{-10}, \sqrt{xp_i^2 + yp_i^2}\right)\right];\ for\ i = 0,1,2,\ldots,n_s - 1.$$

Next, a three-point average is performed on *spectr* to determine the bin, or index i_{peak}, at which the peak spectral amplitude occurs. Once i_{peak} has been determined, the outward propagation angle is calculated as

$$\vartheta_{out} = \mathrm{SIN}^{-1}\left(\frac{\lambda\, i_{peak}}{2 n_s \Delta z_{PE}}\right).$$

Tables 86 and 87 identify, describe, and provide units of measure and computational source for each input and output data element, respectively, of the SPECEST SU.

Table 86. SPECEST SU input data element requirements.

Name	Description	Units	Source
Δz_{PE}	PE mesh height increment (bin width in z-space)	meters	FFTPAR SU
filtp	Array filter for spectral estimation calculations	N/A	APMINIT CSC
jz_{lim}	PE bin # corresponding to z_{lim} (i.e., $z_{lim} = jz_{lim}\, \Delta z_{PE}$)	N/A	APMINIT CSC
ln_p	Power of 2 transform size used in spectral estimation calculations (i.e., $n_p = 2^{lnp}$)	N/A	APMINIT CSC
n_{p34}	¾ n_p	N/A	APMINIT CSC
n_p	Number of bins in upper PE region to consider for spectral estimation.	N/A	APMINIT CSC
n_s	Transform size for spectral estimation calculations	N/A	APMINIT CSC
U	Complex field at current PE range, r	µV/m	PESTEP SU
xo_{con}	Constant used in determining ϑ_{out}	N/A	APMINIT CSC
y_{cur}	Height of ground at current range, r	meters	PESTEP SU

Table 87. SPECEST output data element requirements.

Name	Description	Units
spectr	Spectral amplitude of field	dB
ϑ_{out}	Outward propagation angle at top of PE region	radians
xp	Real part of spectral portion of PE field	µV/m
yp	Imaginary part of spectral portion field	µV/m

5.2.17 Troposcatter (TROPO) SU

The TROPO SU calculates loss due to troposcatter and determines the appropriate loss to add to the already calculated propagation loss at and beyond the radio horizon.

Upon entering the TROPO SU, the current output range, r_{out}, is updated; the tangent angle from the source to the surface, ϑ_1, is initialized to its value for smooth surface, $\vartheta_{1,s}$; and the troposcatter loss term, *tlst*, is initialized to its value for smooth surface, $tlst_s$. If performing a terrain case (f_{ter} = '.true.'), the range from the source to the tangent point, d_1, is initialized, and ϑ_1 is initialized from a previously calculated array of tangent angles to all major terrain peaks. Also, the terrain profile index, j_{t2}, is initialized such that $tx_{j_{t2}-1} \le r_{out} < tx_{j_{t2}}$.

The following steps (1 through 12) are performed for each output height index, j, from j_s to j_e.

1. If running a smooth surface case (f_{ter} = '.false.') and r_{out} is less than the minimum range, rdt_j, at which diffraction field solutions are applicable for the current height, then the SU is exited. Otherwise, the SU program flow continues with step 2.

2. The tangent angle from the receiver, ϑ_2, is initialized to that for the j^{th} receiver height over smooth surface, $\vartheta 2s_j$. However, if f_{ter} = '.true.', then the largest tangent angle, a_2, and range, d_2, from the receiver to the tangent point are determined using an iterative loop performed for index, i, from j_{t2}-1 to 1 in decrements of -1 as follows:

$$r_2 = r_{out} - tx_i$$

$$a_2 = \frac{ty_i - zout_j}{r_2} - \frac{r_2}{2a_{ek}},$$

   where a_{ek} is $^4/_3$ times the earth's radius. If the current ϑ_2 value is less than a_2, then ϑ_2 is set equal to a_2 and d_2 is set equal to r_2. The index i is decremented by one and the above calculations are repeated.

3. Next, if r_{out} is less than the sum of the tangent ranges, d_1 and d_2, then the SU is exited. Otherwise, SU program flow continues with step 4.

4. To account for antenna pattern effects over terrain, the ANTPAT SU is referenced using the tangent angle from the source to determine the antenna pattern factor, $f(\vartheta_1)$. The troposcatter loss term is then adjusted from its smooth surface value as

$$tlst = tlst_s - 20\,\mathrm{LOG}\big[f(\vartheta_1)\big].$$

   The following steps (5 through 12) are now performed regardless of the value of f_{ter}.

5. The common volume scattering angle is given by

$$\theta = \vartheta 0_{i_{stp}} + \vartheta_1 + \vartheta_2.$$

6. Next, the following calculations are made to determine the effective scattering height h_o:

$$a = \frac{1}{2}\vartheta 0_{i_{stp}} + \vartheta_1 + \frac{ant_{ref} - zout_j}{r_{out}}$$
$$b = \frac{1}{2}\vartheta 0_{i_{stp}} + \vartheta_1 - \frac{ant_{ref} - zout_j}{r_{out}},$$

$$s = \text{AMIN}\left[\text{AMAX}\left(.1, \frac{a}{b}\right), 10\right],$$

$$h_o = \frac{s\, r_{out}\, \theta}{10^3 (1+s)^2}.$$

7. The parameter, η_s, is then calculated as a function of h_o:

$$\eta_{sx} = .5696\, h_o \left(1 + s n_1\, e^{-3.8\times 10^{-6} h_o^6}\right)$$
$$\eta_s = \text{AMIN}\left[5, \text{AMAX}(.01, \eta_{sx})\right].$$

8. Next, the parameters, ct_1 and ct_2, are defined as

$$ct_1 = 16.3 + 13.3\eta_s$$
$$ct_2 = .4 + .16\eta_s \quad ,$$

where these are, in turn, used to calculate the quantities r_1 and r_2:

$$r_1 = \text{AMAX}\left(1, r_f\, ant_{ref}\, \theta\right)$$
$$r_2 = \text{AMAX}\left(1, r_f\, zout_j\, \theta\right).$$

The quantity r_f was previously determined by referencing the TROPOINIT SU.

9. The variables, ct_1, ct_2, r_1, and r_2, are next used to determine H_1 and H_2;

$$H_1 = \text{AMAX}\left[0., ct_1 (r_1 + ct_2)^{-\frac{4}{3}}\right]$$
$$H_2 = \text{AMAX}\left[0., ct_1 (r_2 + ct_2)^{-\frac{4}{3}}\right].$$

10. The frequency gain function, H_o, is then determined by

$$H_o = \frac{H_1 + H_2}{2} + \Delta H_o,$$

where

$$\Delta H_o = 6\left(.6 - \text{LOG}(\eta_S)\right)\text{LOG}(s)\,\text{LOG}(q_l),$$

and

$$q_t = \mathrm{AMIN}\left[10., \mathrm{AMAX}\left(.1, \frac{r_2}{s r_1}\right)\right].$$

ΔH_o is not allowed to be larger than ½(H_1+ H_2), and H_o is set equal to 0 if it becomes negative.

11. Next, the troposcatter loss is computed from

$$t_{loss} = tlst + 573\theta + rlogo_{i_{sp}} + H_o .$$

12. Finally, troposcatter loss is compared to propagation loss. If the difference between the propagation loss and troposcatter loss is less than 18 dB, the corresponding power levels of the two loss values are added. If the difference is greater than 18 dB, the lesser of the two losses is used. Resulting loss is given by

$$rloss_j = t_{loss} \qquad for \;\; rloss_j - t_{loss} \geq 18$$

$$rloss_j = rloss_j - 10\mathrm{LOG}\left(1 + 10^{.1\left(rloss_j - t_{loss}\right)}\right) \quad for \;\; rloss_j - t_{loss} \geq -18 .$$

Tables 88 and 89 identify, describe, and provide units of measure and computational source for each input and output data element of the TROPO SU.

Table 88. TROPO SU input data element requirements.

Name	Description	Units	Source
adl	Array of tangent ranges from source height—used with terrain profile	meters	TROPOINIT SU
$adif$	Height differences between ant_{ref} and all output receiver heights	meters	TROPOINIT SU
a_{ek2}	Twice $^4/_3$ effective earth's radius	meters	APMINIT CSC
$d2s$	Array of tangent ranges for all output receiver heights over smooth surface	meters	TROPOINIT SU
e_k	$^4/_3$ effective earth's radius factor	N/A	APM_MOD
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
i_{sp}	Current output range step index	N/A	Calling SU
i_{tpa}	Number of height/range points pairs in profile, tx, ty	N/A	APMINIT CSC

Table 88. TROPO SU input data element requirements. (Continued)

Name	Description	Units	Source
j_e	Ending receiver height index at which to compute troposcatter loss	N/A	Calling SU
j_s	Starting receiver height index at which to compute troposcatter loss	N/A	Calling SU
j_{t1}	Index counter for $ad1$ and $\vartheta 1t$ arrays	N/A	TROPOINIT SU APMSTEP CSC
j_{t2}	Index counter for tx and ty arrays indicating location of receiver range	N/A	TROPOINIT SU APMSTEP CSC
ktr_1	Number of tangent ranges from source height	N/A	TROPOINIT SU
rdt	Array of minimum ranges at which diffraction field solutions are applicable (for smooth surface) for all output receiver heights.	meters	TROPOINIT SU
r_f	Constant used for troposcatter calculations	meters^{-1}	TROPOINIT SU
$rlogo$	Array containing 20 times the logarithm of all output ranges	N/A	APMINIT CSC
$rloss$	Propagation loss	dB	Calling SU
rt_1	$r_f * ant_{ref}$	N/A	TROPOINIT SU
$rngout$	Array containing all desired output ranges	meters	APMINIT CSC
sn_1	Term used in troposcatter loss calculation	N/A	TROPOINIT SU
$\vartheta 0$	Array of angles used to determine common volume scattering angle	radians	TROPOINIT SU
ϑ_{1s}	Tangent angle from source (for smooth surface)	radians	TROPOINIT SU
$\vartheta 2s$	Array of tangent angles from all output receiver heights—used with smooth surface	radians	TROPOINIT SU
$\vartheta 1t$	Array of tangent angles from source height—used with terrain profile	radians	TROPOINIT SU
$tlst_s$	Troposcatter loss term for smooth surface case	dB	TROPOINIT SU
tx	Range points of terrain profile	meters	TERINIT SU
ty	Adjusted height points of terrain profile	meters	TERINIT SU
$zout$	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 89. TROPO SU output data element requirements.

Name	Description	Units
j_{t1}	Index counter for $ad1$ and $\vartheta t1$ arrays	N/A
j_{t2}	Index counter for tx and ty arrays indicating location of receiver range	N/A
rloss	Propagation loss	dB

5.3 EXTENDED OPTICS INITIALIZATION (XOINIT) CSC

The XOINIT CSC initializes the range, height, and angle arrays in preparation for XO calculations.

Upon entering the XOINIT CSC, the value of i_{xostp} is tested. If i_{xostp} is equal to 0, then the CSC is exited. If i_{xostp} is greater than 0, then the following procedure is performed.

The arrays *curang* and *curng*, used for storage of traced local angles and ranges, respectively, are allocated and initialized to the range and angle values stored in *ffacz*. The array *curht* is allocated and initialized to the height of the top of the PE calculation region, z_{lim}. The arrays *igrd*, *htout*, and *prfac*, used for storage of starting refractivity gradient level (at which to begin ray tracing), final output heights, and propagation factors at a particular range, respectively, are also allocated and initialized to 0. Next, the dummy array, *dum*, used for temporary storage, is allocated and initialized to 0.

If f_{ter} is '.true.', then the SMOOTH SU is referenced twice to perform a 10-point smoothing operation on the angle values, using *dum* for temporary storage of angles after the first pass smoothing operation.

Next, the starting height index at which to begin XO calculations, j_{xstart} is initialized to the ending height index for PE calculations, j_{end}, plus one. Finally, *dum* is deallocated before exiting.

Tables 90 and 91 identify, describe, and provide units of measure and computational source for each input and output data element of the XOINIT CSC.

Table 90. XOINIT CSC input data element requirements.

Name	Description	Units	Source
ffacz	Array containing propagation factor, range, and propagation angle at z_{lim}	dB, meters, radians	FZLIM SU
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU

Table 90. XOINIT CSC input data element requirements. (Continued)

Name	Description	Units	Source
i_{xostp}	Current output range step index for XO calculations	N/A	Calling SU
iz	Number of propagation factor, range, angle triplets stored in *ffacz*	N/A	APMINIT CSC FZLIM SU
j_{end}	Ending index within *mloss* of PE loss values	N/A	Calling SU
z_{lim}	Height limit for PE calculation region	meters	GETTHMAX SU

Table 91. XOINIT CSC output data element requirements.

Name	Description	Units
curang	Array of current local angles for each ray being traced in XO region	Radians
curht	Array of current local heights for each ray being traced in XO region	meters
curng	Array of current local ranges for each ray being traced in XO region	meters
htout	Final height for each ray traced in XO region at range, r_{out}	meters
i_{error}	Error flag	N/A
igrd	Integer indexes indicating at what refractive gradient level to begin ray tracing for next XO range step for each ray in XO region.	N/A
j_{xstart}	Starting index within *mloss* of XO loss values	N/A
prfac	Propagation factor for each ray traced in XO region range, r_{out}	dB

5.3.1 Smooth (SMOOTH) SU

The SMOOTH SU performs a i_{av}-point average smoothing operation on the array passed to it.

The array *arbef* is passed to the SU, along with the number of points over which to perform the smoothing operation, i_{av}. Once the smoothing operation has been performed, the resulting "smoothed" points are stored in *arbef* and passed back to the calling routine.

Tables 92 and 93 identify, describe, and provide units of measure and computational source for each input and output data element of the SMOOTH SU.

Table 92. SMOOTH SU input data element requirements.

Name	Description	Units	Source
arbef	Array of angles before smoothing operation	radians	Calling SU
i_{av}	Number of points over which to perform average smoothing	N/A	Calling SU
iz	Number of propagation factor, range, angle triplets stored in *ffacz*	N/A	Calling SU

Table 93. SMOOTH SU output data element requirements.

Name	Description	Units
araft	Array of angles after smoothing operation	radians

5.4 EXTENDED OPTICS STEP (XOSTEP) CSC

The XOSTEP CSC calculates the propagation loss in the XO region for one output range step.

Upon entering the XOSTEP CSC, the current execution mode is checked to determine if XO calculations will be necessary ($i_{hybrid} \neq 0$). If i_{hybrid} is 0, then the CSC is exited.

If i_{hybrid} is not equal to 0, the output range, r_{out}, and the square of the output range, r_{sq}, are updated. The *mloss* values are initialized to -1 from the index of the start of XO calculations, j_{xstart}, to the maximum number of height output points, n_{zout}. The EXTO SU is then referenced to calculate propagation loss values in the XO region. Loss values are returned from $mloss_{j_{xstart}}$ to $mloss_{j_{xe}}$.

If FE and RO calculations need to be performed ($i_{hybrid} = 1$), then the indices j_{fs} and j_{fe}, indicating the height index at which to start and end FE calculations, respectively, are determined. The FEM SU is then referenced to compute propagation loss values from $mloss_{j_{fs}}$ to $mloss_{j_{fe}}$. Similarly for RO calculations, the indices, j_{rs}, and j_{re} are determined, and the ROM SU is referenced to compute propagation loss values from $mloss_{j_{rs}}$ to $mloss_{j_{re}}$.

Finally, the index, j_{xend}, is set equal to the maximum of j_{xe}, j_{fe}, and n_{zout}. If absorption loss needs to be calculated (k_{abs}>0), then loss due to gaseous absorption is computed and added to propagation loss values from $mloss_{j_{xstart}}$ to $mloss_{j_{xend}}$.

Tables 94 and 95 identify, describe, and provide units of measure and computational source for each input and output data element of the XOSTEP CSC.

Table 94. XOSTEP CSC input data element requirements.

Name	Description	Units	Source
gas_{att}	Gaseous absorption attenuation rate	dB/km	GASABS SU
htfe	Array containing the height at each output range separating the FE region from the RO region (full hybrid mode), or the FE region from the PE region (partial hybrid mode)	meters	FILLHT SU
ht_{lim}	Maximum height relative to h_{minter}	meters	TERINIT SU
i_{hybrid}	Integer indicating which sub-models will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	N/A	GETMODE SU
i_{stp}	Current output range step index	N/A	Calling CSCI
j_{xstart}	Index at which valid loss values in *mloss* start	N/A	Calling CSCI
k_{abs}	Gaseous absorption calculation flag: $k_{abs} = 0$; no absorption loss $k_{abs} = 1$; compute absorption loss based on air temperature, t_{air}, and absolute humidity, abs_{hum} $k_{abs} = 2$; compute absorption loss based on specified absorption attenuation rate, γ_a	N/A	APMINIT CSC
n_{zout}	Integer number of output height points desired	N/A	Calling CSCI
rngout	Array containing all desired output ranges	meters	APMINIT CSC
rsqrd	Array containing the square of all desired output ranges	meters^2	APMINIT CSC
zout	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 95. XOSTEP CSC output data element requirements.

Name	Description	Units
j_{xend}	Index at which valid loss values in *mloss* end	N/A
mloss	Propagation loss array	cB
r_{out}	Current desired output range	meters

5.4.1 Extended Optics (EXTO) SU

The EXTO SU calculates propagation loss based on XO techniques. The SU performs a ray trace on all rays within one output range step and returns the propagation loss up to the necessary height, storing all angle, height, and range information for subsequent ray tracing upon the next reference to the SU.

Upon entering the SU, internal one-line ray trace functions are defined as

$$\mathrm{RADA1}(a,b)=a^2+2\,g_{rd}\,b$$
$$\mathrm{RP}(a,b)=a+\frac{b}{g_{rd}}$$
$$\mathrm{AP}(a,b)=a+b\,g_{rd}$$
$$\mathrm{HP}(a,b)=a+\frac{b^2-c^2}{2\,g_{rd}}$$

and a one-line interpolation function is also defined as

$$\mathrm{PLINT}(pl_1,pl_2,f_{rac})=pl_1+f_{rac}\left(pl_2-pl_1\right).$$

Next, several variables are initialized: the free-space loss at the current output range, fsl_{rout}, is updated; the starting and ending index counters, iz_s and iz_e, for the local angle, range, and height arrays are initialized to 1 for the first reference to the EXTO SU; and the refractivity profile starting index, i_{rps}, is also initialized to 1 for the first reference to the EXTO SU. The index, iz_e, is then determined such that $curng_{iz_e} \le r_{out} < curng_{iz_e+1}$.

The following ray trace steps (1 through 3) are performed for each ray (i.e., each j^{th} angle, range, and height triplet, for j ranging from iz_s to iz_e)

1. At the start of the ray trace, the current local angle, (a_0), range, (r_0), height, (h_0), and refractive gradient index, (i_{grad}), are initialized to $curang_j$, $curng_j$, $curht_j$, and $igrd_j$, respectively. Next, refractive profile index, i_{rp}, is initialized to the maximum of j or i_{rps}. Finally, the refractivity gradient, g_{rd}, is set equal to the gradient at the $i_{grad}{}^{th}$ level of the $i_{rp}{}^{th}$ profile, $grad_{i_{grad},i_{rp}}$. Next, the following ray trace steps (a through d) are performed until the current local range, r_0, becomes greater than or equal to r_{out}.

    a. The ending range, r_1, in the ray trace segment is set equal to the minimum of $ffacz_{2,i_{rp}+1}$ or r_{out}. If i_{rp} is equal to the number of stored triplets, iz, then r_1 is set equal to r_{out}.

    b. The jth ray is then traced to r1 and the resulting angle and height at the end of the segment is determined via the in-line functions as

$$a_1=\mathrm{AP}(a_0,r_1-r_0)$$
$$h_1=\mathrm{HP}(h_0,a_1,a_0)\,.$$

c. The ending height h_1 is then compared to the next height level in the current refractivity profile, $htr_{i_{grad}+1,i_{rp}}$ and if h_1 is greater than this height level, it is set equal to $htr_{i_{grad}+1,i_{rp}}$ and a new a_1 and r_1 are computed from

$$a_1 = \sqrt{\mathrm{RADA1}(a_0, h_1 - h_0)}$$
$$r_1 = \mathrm{RP}(r_0, a_1 - a_0)$$

i_{grad} is then set to the minimum of i_{grad}+1 or $lvl_{i_{rp}}$-1.

d. The starting angle, range, and height for the next ray trace segment is updated, and if necessary, the refractivity profile index i_{rp} is updated to the minimum of i_{rp}+1 or iz_e. Steps 1a through 1d are then repeated for the next ray segment.

2. Once the ray has been traced to a range of r_{out} or greater, the current angle, range, and height arrays, *curang, curng,* and *curht*, respectively, are updated to the values for a_0, r_0, and h_0 for subsequent references to the EXTO SU.

3. The counter, *k*, for the propagation factor array, *prfac*, and corresponding height array, *htout*, is incremented by one and the arrays are updated according to

$$prfac_k = ffacz_{1,j}$$
$$htout_k = h_0$$

Once all rays have been traced, the starting profile index, i_{rps}, is updated to iz_e for the next reference to the EXTO SU, and the counter, *k*, is again incremented by one and the last value of *prfac* and *htout* are updated as follows,

$$prfac_k = ffrout_{1,i_{sp}}$$
$$htout_k = ffrout_{2,i_{sp}}$$

The number of traced XO height points, n_{xo}, at the current output range is then set to *k*. Note that at this point, all output heights in *htout* are decreasing from $htout_1$ to $htout_{n_{xo}}$ and all traced heights in *curht* are decreasing from $curht_{iz_s}$ to $curht_{iz_e}$.

The starting index, iz_s, is then adjusted (for the next reference to the EXTO SU) if the topmost traced height, $curht_{iz_s}$, is greater than ht_{lim}. If performing a terrain case, the output height points may not be continually decreasing from $htout_1$ to $htout_{n_{xo}}$. In this case, *htout* is sorted, along with *prfac*, such that all height values are steadily decreasing. The ending index, j_{xe}, at which XO loss values will be calculated and stored in *mloss*, is set equal to n_{zout} and adjusted, if necessary, such that $zout_{j_{xe}}$ is less than $htout_1$. Now, the counter index, *ix*, is initialized to n_{xo}. Next, the propagation loss values are determined via linear interpolation on the values in *prfac*. The following steps (1 through 3) are performed for each output height point $zout_j$ from j_{xs} to j_{xe}.

1. The counter *ix* is adjusted (if necessary) such that $htout_{ix} \le zout_j < htout_{ix-1}$.

2. The propagation factor, F_{fac}, at height, $zout_j$, is then calculated according to

$$F_{fac} = \mathrm{PLINT}(prfac_{ix}, prfac_{ix-1}, f_{rac}),$$

where

$$f_{rac} = \frac{zout_j - htout_{ix}}{htout_{ix-1} - htout_{ix}} .$$

3. The propagation loss is now calculated as

$$rloss_j = F_{fac} + fsl_{rout} .$$

Once all propagation loss values have been computed, the TROPO SU is referenced to compute troposcatter loss, if necessary. Finally, the loss is converted to centibels and stored in *mloss* before exiting.

Tables 96 and 97 identify, describe, and provide units of measure and computational source for each input and output data element of the EXTO SU. Table 101 identifies terms that are used internal to the EXTO SU and whose value must be retained from SU call to SU call for reasons of computational efficiency.

Table 96. EXTO SU input data element requirements.

Name	Description	Units	Source
curang	Array of current local angles for each ray being traced in XO region	radians	EXTO SU XOINIT CSC
curht	Array of current local heights for each ray being traced in XO region	meters	EXTO SU XOINIT CSC
curng	Array of current local ranges for each ray being traced in XO region	meters	EXTO SU XOINIT CSC
ffacz	Array containing propagation factor, range, and propagation angle at z_{lim}	dB, meters, radians	FZLIM SU
ffrout	Array of propagation factors at each output range beyond r_{atz} and at height, z_{lim}	dB	CALCLOS SU
fslr	Free-space loss array for output ranges	dB	APMINIT CSC
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	N/A	TERINIT SU
grad	Two-dimensional array containing gradients of each profile used in XO calculations	M-units /meter	SAVEPRO SU

Table 96. EXTO SU input data element requirements. (Continued)

Name	Description	Units	Source
hlim	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	meters	GETTHMAX SU
ht_{lim}	Maximum height relative to h_{minter}	meters	TERINIT SU
htr	Two-dimensional array containing heights of each profile used in XO calculations	meters	SAVEPRO SU
igrd	Integer indexes indicating at what refractive gradient level to begin ray tracing for next XO range step for each ray in XO region.	N/A	XOINIT CSC
i_{ratz}	Index of output range step in which to begin storing propagation factor and outgoing angle for XO region	N/A	APMINIT CSC
i_{stp}	Current output range step index	N/A	Calling SU
i_{tropo}	Troposcatter calculation flag: $i_{tropo} = 0$; no troposcatter calcs $i_{tropo} = 1$; troposcatter calcs	N/A	Calling CSCI
iz	Number of propagation factor, range, and angle triplets stored in *ffacz*	N/A	FZLIM SU
j_{xs}	Index at which valid loss values in *mloss* start	N/A	Calling SU
lvl	Number of height levels in each profile used in XO calculations	N/A	SAVEPRO SU
n_{zout}	Integer number of output height points desired	N/A	Calling CSCI
r_{out}	Current output range	meters	Calling SU
zout	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 97. EXTO SU output data element requirements.

Name	Description	Units
curang	Array of current local angles for each ray being traced in XO region	radians
curht	Array of current local heights for each ray being traced in XO region	meters
curng	Array of current local ranges for each ray being traced in XO region	meters
hlim	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	meters

Table 97. EXTO SU output data element requirements. (Continued)

Name	Description	Units
htout	Final height for each ray traced in XO region at range, r_{out}	meters
j_{xe}	Index at which valid loss values in *mloss* end	N/A
mloss	Propagation loss array	cB
prfac	Propagation factor for each ray traced in XO region range, r_{out}	dB
rloss	Propagation loss	dB

Table 98. EXTO SU save data element requirements.

Name	Description	Units
i_{rps}	Starting index counter for refractivity profiles	N/A
iz_e	Ending index in *curang*, *curng*, and *curht* to trace to r_{out}	N/A
iz_s	Starting index in *curang*, *curng*, and *curht* to trace to r_{out}	N/A

5.5 APMCLEAN CSC

The APMCLEAN CSC deallocates all dynamically dimensioned arrays used in one complete run of APM calculations. Upon entry, all arrays that were dynamically allocated at the beginning of the current application are now deallocated.

Tables 99 and 100 identify, describe, and provide units of measure and computational source for each input and output data element of the APMCLEAN CSC.

Table 99. APMCLEAN CSC input data element requirements.

Name	Description	Units	Source
*ad*1	Array of tangent ranges from source height—used with terrain profile	meters	TROPOINIT SU
adif	Height array used for troposcatter calcs	meters	TROPOINIT SU
curang	Array of current local angles for each ray being traced in XO region	radians	EXTO SU XOINIT CSC
curht	Array of current local heights for each ray being traced in XO region	meters	EXTO SU XOINIT CSC
curng	Array of current local ranges for each ray being traced in XO region	meters	EXTO SU XOINIT CSC

Table 99. APMCLEAN CSC input data element requirements. (Continued)

Name	Description	Units	Source
d2s	Array of tangent ranges for all output receiver heights over smooth surface	meters	TROPOINIT SU
dielec	Two-dimensional array containing the relative permittivity and conductivity; $dielec_{1,i}$ and $dielec_{2,i}$, respectively.	N/A, S/m	Calling CSCI, DIEINIT SU
envpr	Complex (refractivity) phase term array interpolated every Δz_{PE} in height	N/A	PHASE2 SU
ffacz	Array containing propagation factor, range, and propagation angle at z_{lim}	dB, meters, radians	FZLIM SU
ffrout	Array of propagation factors at each output range beyond r_{atz} and at height, z_{lim}	dB	CALCLOS SU
filt	Cosine-tapered (Tukey) filter array	N/A	APMINIT CSC
filtp	Array filter for spectral estimation calculations	N/A	APMINIT CSC
frsp	Complex free-space propagator term array	N/A	PHASE1 SU
fslr	Free-space loss array for output ranges	dB	APMINIT CSC
gr	Intermediate M-unit gradient array, RO region	(M-unit/ m)10^{-6}	REFINIT SU
grad	Two-dimensional array containing gradients of each profile used in XO calculations	M-units /meter	SAVEPRO SU
hfangr	Array of user-defined cut-back angles. This is used only for user-defined height-finder antenna type.	radians	APMINIT CSC
hlim	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	meters	GETTHMAX SU
href	Heights of refractivity profile with respect to y_{ref}	meters	PROFINT SU
ht	PE mesh height array of size, n_{fft}	meters	APMINIT CSC
htdum	Height array for current interpolated profile	meters	REFINIT SU REFINTER SU
htfe	Array containing the height at each output range separating the FE region from the RO region (full hybrid mode), or the FE region from the PE region (partial hybrid mode)	meters	FILLHT SU
htout	Final height for each ray traced in XO region at range r_{out}	meters	XOINIT CSC
htr	Two-dimensional array containing heights of each profile used in XO calculations	meters	SAVEPRO SU

Table 99. APMCLEAN CSC input data element requirements. (Continued)

Name	Description	Units	Source
igrd	Integer indexes indicating at what refractive gradient level to begin ray tracing for next XO range step for each ray in XO region.	N/A	XOINIT CSC
igrnd	Integer array containing ground type composition for given terrain profile—can vary with range. Different ground types are: 0 = Seawater 1 = Freshwater 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1°C 6 = Ice at -10°C 7 = User-defined (in which case, values of relative permittivity and conductivity must be given).	N/A	Calling CSCI
lvl	Number of height levels in each profile used in XO calculations	N/A	SAVEPRO SU
nc^2	Array of complex dielectric constants	N/A	DIEINIT SU
prfac	Propagation factor for each ray traced in XO region range, r_{out}	dB	XOINIT CSC
profint	Profile interpolated to every Δz_{PE} in height	M-units	REFINTER SU
q	Intermediate M-unit difference array, RO region	2M 10^{-6}	REFINIT SU
rdt	Array of minimum ranges at which diffraction field solutions are applicable (for smooth surface) for all output receiver heights.	meters	TROPOINIT SU
refdum	M-unit array for current interpolated profile	M-units	REFINIT SU REFINTER SU
refref	Refractivity profile with respect to y_{ref}	M-units	PROFINT SU
*rfac*1	Propagation factor at valid output height points from PE field at range, r_{last}.	dB	CALCLOS SU
*rfac*2	Propagation factor at valid output height points from PE field at range, r	dB	CALCLOS SU
rgrnd	Array containing ranges at which varying ground types apply.	meters	Calling CSCI

Table 99. APMCLEAN CSC input data element requirements. (Continued)

Name	Description	Units	Source
rlogo	Array containing 20 times the logarithm of all output ranges	N/A	APMINIT CSC
rloss	Propagation loss	dB	ALLARRAY_APM CALCLOS SU EXTO SU TROPO SU
rm	Intermediate M-unit array, RO region	M 10^{-6}	REFINIT SU
rngout	Array containing all desired output ranges	meters	APMINIT CSC
root	Array of R_T to the i^{th} power (e.g., $root_i = R_T^i$)	N/A	GETALN SU
rootm	Array of $-R_T$ to the i^{th} power (e.g., $rootm_i = (-R_T)^i$)	N/A	GETALN SU
rsqrd	Array containing the square of all desired output ranges	meters^2	APMINIT CSC
slp	Slope of each segment of terrain	N/A	TERINIT SU
ϑ0	Array of angles used to determine common volume scattering angle	radians	TROPOINIT SU
ϑ2s	Array of tangent angles from all output receiver heights—used with smooth surface	radians	TROPOINIT SU
ϑ1t	Array of tangent angles from source height - used with terrain profile	radians	TROPOINIT SU
tx	Range points of terrain profile	meters	TERINIT SU
ty	Adjusted height points of terrain profile	meters	TERINIT SU
U	Complex PE field	µV/m	PESTEP SU
Ulast	Complex PE field at range, r_{last}	µV/m	PESTEP SU
w	Difference equation of complex PE field	µV/m^2	PESTEP SU
xdum	Real part of complex field array	µV/m	FFT SU
xp	Real part of spectral portion of PE field	µV/m	SPECEST SU
ydum	Imaginary part of complex field array	µV/m	FFT SU
ym	Particular solution of difference equation	µV/m	PESTEP SU
yp	Imaginary part of spectral portion field	µV/m	SPECEST SU
zout	Array containing all desired output heights referenced to h_{minter}	meters	APMINIT CSC

Table 99. APMCLEAN CSC input data element requirements. (Continued)

Name	Description	Units	Source
$zoutma_j$	j^{th} output height point relative to "real" ant_{ref}	meters	APMINIT CSC
$zoutpa_j$	j^{th} output height point relative to "image" ant_{ref}	meters	APMINIT CSC
zRO	Array of output heights in RO region	meters	APMINIT CSC
zrt	Intermediate height array, RO region	meters	REFINIT SU

Table 100. APMCLEAN CSC output data element requirements.

Name	Description	Units
i_{error}	Error flag indicator: non-zero if error has occured in deallocation procedure	N/A

6. REQUIREMENTS TRACEABILITY

This section provides the traceability of the design of the APM CSCI. Table 101 presents this traceability between the corresponding sections of the Software Requirements Specification (SRS) and the Software Design Description (SDD) and between the various components of the APM CSCI.

Table 101. Traceability Matrix between the SRS and the SDD.

Software Requirements Specification		Software Design Description	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
CSCI Capability Requirements	3.1	CSCI-WIDE DESIGN DECISIONS	3
CSCI Capability Requirements	3.1	CSCI Components	4.1
CSCI Capability Requirements	3.1	Concept of Execution	4.2
Advance Propagation Initialization (APMINIT) CSC	3.1.1	Advance Propagation Initialization (APMINIT) CSC	5.1
Allocate Arrays APM (ALLARRAY_APM) SU	3.1.1.1	Allocate Arrays APM (ALLARRAY_APM) SU	5.1.1
Allocate Array PE (ALLARRAY_PE) SU	3.1.1.2	Allocate Array PE (ALLARRAY_PE) SU	5.1.2
Allocate Array XO (ALLARRAY_XO) SU	3.1.1.3	Allocate Array XO (ALLARRAY_XO) SU	5.1.3

Table 101. Traceability Matrix between the SRS and the SDD. (Continued)

Software Requirements Specification		Software Design Description	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
Antenna Pattern (ANTPAT) SU	3.1.1.4	Antenna Pattern (ANTPAT) SU	5.1.4
Dielectric Initialization (DIEINIT) SU	3.1.1.5	Dielectric Initialization (DIEINIT) SU	5.1.5
Fast-Fourier Transform (FFT) SU	3.1.1.6	Fast-Fourier Transform (FFT) SU	5.1.6
FFT Parameters (FFTPAR) SU	3.1.1.7	FFT Parameters (FFTPAR) SU	5.1.7
Fill Height Arrays (FILLHT) SU	3.1.1.8	Fill Height Arrays (FILLHT) SU	5.1.8
Gaseous Absorption (GASABS) SU	3.1.1.9	Gaseous Absorption (GASABS) SU	5.1.9
Get Alpha Impedance (GETALN) SU	3.1.1.10	Get Alpha Impedance (GETALN) SU	5.1.10
Get Mode (GETMODE) SU	3.1.1.11	Get Mode (GETMODE) SU	5.1.11
Get Maximum Angle (GETTHMAX) SU	3.1.1.12	Get Maximum Angle (GETTHMAX) SU	5.1.12
Interpolate Profile (INTPROF) SU	3.1.1.13	Interpolate Profile (INTPROF) SU	5.1.13
Free-Space Propagator Phase Term (PHASE1) SU	3.1.1.14	Free-Space Propagator Phase Term (Phase1) SU	5.1.14
Environmental Propagator Phase Term (PHASE2) SU	3.1.1.15	Environmental Propagator Phase Term (Phase2) SU	5.1.15
Profile Reference (PROFREF) SU	3.1.1.16	Profile Reference (PROFREF) SU	5.1.16
Refractivity Initialization (REFINIT) SU	3.1.1.17	Refractivity Initialization (REFINIT) SU	5.1.17
Sine Fast-Fourier Transform (SINFFT) SU	3.1.1.18	Sine Fast-Fourier Transform (SINFFT) SU	5.1.18
Terrain Initialization (TERINIT) SU	3.1.1.19	Terrain Initialization (TERINIT) SU	5.1.19
Troposcatter Initialization (TROPOINIT) SU	3.1.1.20	Troposcatter Initialization (TROPOINIT) SU	5.1.20
Starter Field Initialization (XYINIT) SU	3.1.1.21	Starter Field Initialization (XYINIT) SU	5.1.21
Advance Propagation Model Step (APMSTEP) CSC	3.1.2	Advance Propagation Model Step (APMSTEP) CSC	5.2
Calculate Propagation Loss (CALCLOS) SU	3.1.2.1	Calculate Propagation Loss (CALCLOS) SU	5.2.1
DoShift SU	3.1.2.2	DoShift SU	5.2.2
Flat Earth Model (FEM) SU	3.1.2.3	Flat Earth Model (FEM) SU	5.2.3

Table 101. Traceability Matrix between the SRS and the SDD. (Continued)

Software Requirements Specification		Software Design Description	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
Free-Space Range Step (FRSTP) SU	3.1.2.4	Free-Space Range Step (FRSTP) SU	5.2.4
FZLIM SU	3.1.2.5	FZLIM SU	5.2.5
Get Propagation Factor (GETPFAC) SU	3.1.2.6	Get Propagation Factor (GETPFAC) SU	5.2.6
Get Reflection Coefficient (GETREFCOEF) SU	3.1.2.7	Get Reflection Coefficient (GETREFCOEF) SU	5.2.7
Parabolic Equation Step (PESTEP) SU	3.1.2.8	Parabolic Equation Step (PESTEP) SU	5.2.8
Ray Trace (RAYTRACE) SU	3.1.2.9	Ray Trace (RAYTRACE) SU	5.2.9
Refractivity Interpolation (REFINTER) SU	3.1.2.10	Refractivity Interpolation (REFINTER) SU	5.2.10
Remove Duplicate Refractivity Levels (REMDUP) SU	3.1.2.11	Remove Duplicate Refractivity Levels (REMDUP) SU	5.2.11
Ray Optics Calculation (ROCALC) SU	3.1.2.12	Ray Optics Calculation (ROCALC) SU	5.2.12
Ray Optics Loss (ROLOSS) SU	3.1.2.13	Ray Optics Loss (ROLOSS) SU	5.2.13
Ray Optics Model (ROM) SU	3.1.2.14	Ray Optics Model (ROM) SU	5.2.14
Save Profile (SAVEPRO) SU	3.1.2.15	Save Profile (SAVEPRO) SU	5.2.15
Spectral Estimation (SPECEST) SU	3.1.2.16	Spectral Estimation (SPECEST) SU	5.2.16
Troposcatter (TROPO) SU	3.1.2.17	Troposcatter (TROPO) SU	5.2.17
Extended Optics Initialization (XOINIT) CSC	3.1.3	Extended Optics Initialization (XOINIT) CSC	5.3
Smooth (SMOOTH) SU	3.1.3.1	Smooth (SMOOTH) SU	5.3.1
Extended Optics Step (XOSTEP) CSC	3.1.4	Extended Optics Step (XOSTEP) CSC	5.4
Extended Optics (EXTO) SU	3.1.4.1	Extended Optics (EXTO) SU	5.4.1
Advanced Propagation Model Clean (APMCLEAN) CSC	3.1.5	Advanced Propagation Model Clean (APMCLEAN) CSC	5.5
CSCI External Interface Requirements	3.2	External Interface	4.3.2
CSCI Internal Interface Requirements	3.3	Internal Interface	4.3.3

Table 101. Traceability Matrix between the SRS and the SDD. (Continued)

Software Requirements Specification		Software Design Description	
SRS Requirement Name	SRS Paragraph Number	Software Design Description Name	SDD Paragraph Number
CSCI Internal Data Requirements	3.4	Internal Data	4.3.4
Environmental Radio Refractivity field Data Element	3.5.1	Environmental Radio Refractivity field Data Element	7.2
Terrain Profile Data Element	3.5.2	Terrain Profile Data Element	7.3
Implementation and Application Considerations	3.10.1	Implementation and Application Considerations	7.1

7. NOTES

7.1 APM CSCI IMPLEMENTATION AND APPLICATION CONSIDERATIONS

The calling TESS-NC CSCI application will determine the employment of the APM CSCI. However, the intensive computational nature of the APM CSCI must be considered when designing an efficient calling application. For this reason, the APM CSCI is designed with flexibility for various hardware suites and computer resource management considerations. This APM CSCI applies only to a coverage and loss diagram application. The following highly recommended guidelines are provided to aid in the design of a coverage or loss diagram application that will most efficiently employ the APM CSCI.

The APM CSCI propagation loss calculations are independent of any target or receiver considerations, therefore, for any EM emitter, one execution of the APM CSCI may be used to create both a coverage diagram and a loss diagram. Since both execution time and computer memory allocation should be a consideration when employing this model, it is most efficient and appropriate to execute the APM CSCI for a particular EM system/environmental/terrain combination before executing any application. The output of the APM CSCI would be stored in a file which would be accessed by multiple applications.

For example, the TESS-NC operator may desire a coverage diagram for one particular radar system. At the beginning of the coverage diagram application, a check would be made for the existence of a previously created APM CSCI output file appropriate for the EM system, environmental, and terrain conditions. If such a file exists, the propagation loss values would be read from the file and used to create the coverage diagram. If the file does not exist, the APM CSCI would be executed to create one. As the APM CSCI is executing, its output could be routed simultaneously to a graphics display device and a file. This file could then be used in the loss diagram application should the operator also choose it. Two distinct applications, therefore, are achieved with only one execution of the APM CSCI. Additionally, should the operator desire an individual coverage diagram for each of multiple targets, or a single coverage diagram illustrating radar detection of a low-flying missile superimposed upon a coverage diagram illustrating his own radar's vulnerability as defined by the

missile's ESM receiver, only a single execution of the APM CSCI would be required, thereby saving valuable computer resources.

7.2 ENVIRONMENTAL RADIO REFRACTIVITY FIELD DATA ELEMENTS

The radio-refractivity field (i.e. the profiles of M-units versus height) should consist of vertical piece-wise linear profiles specified by couplets of height in meters with respect to mean sea level and modified refractivity (M-units) at multiple arbitrary ranges. All vertical profiles must contain the same number of vertical data points, and be specified such that each numbered data point corresponds to like-numbered points (i.e. features) in the other profiles. The first numbered data point of each profile must correspond to a height of zero mean sea level and the last numbered data point must correspond to a height such that the modified refractivity for all greater heights is well represented by extrapolation using the two highest profile points specified.

With the inclusion of terrain and allowing the terrain profile to fall below mean sea level, refractivity profiles can also be provided in which the first level is less than 0 (or below mean sea level). For a terrain profile that falls below mean sea level at some point, the assumption is that the minimum height may be less than the first height in any refractivity profile specified. Therefore, an extrapolation flag, i_{extra}, must be specified to indicate how the APM CSCI should extrapolate from the first refractivity level to the minimum height along the terrain profile. Setting i_{extra} to 0 will cause the APM CSCI to extrapolate to the minimum height using a standard atmosphere gradient; setting i_{extra} to 1 will cause the APM CSCI to extrapolate to the minimum height using the gradient determined from the first two levels of the refractivity profile.

Within each profile, each numbered data point must correspond to a height greater than or equal to the height of the previous data point. Note that this requirement allows for a profile that contains redundant data points. Note also that all significant features of the refractivity profiles must be specified, even if they are above the maximum output height specified for a particular application of APM.

The TESS-NC CSCI application designer and the TESS-NC operator share responsibility for determining appropriate environmental inputs. For example, a loss diagram may be used to consider a surface-to-surface radar detection problem. Since the operator is interested in surface-to-surface, he may truncate the profile, assuming that effects from elevated ducting conditions are negligible. It may be however, that the elevated duct does indeed produce a significant effect. The operator should ensure, therefore, that the maximum height of the profile allows for the inclusion of all significant refractive features.

This specification allows a complicated refractivity field to be described with a minimum of data points. For example, a field in which a single trapping layer linearly descends with increasing range can be described with just two profiles containing only four data points each, frame (a) of figure 3. In the same manner, other evolutions of refractive layers may be described. Frames (b) and (c) of figure 3 show two possible scenarios for the development of a trapping layer. The scenario of choice is the one that is consistent with the true thermodynamical and hydrological layering of the atmosphere.

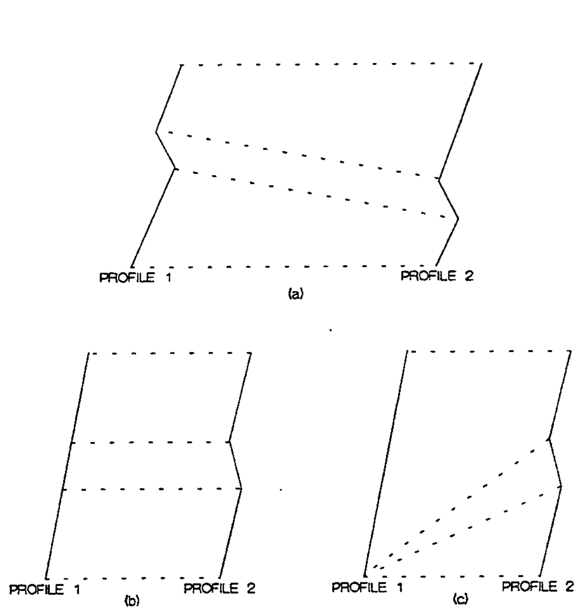

Figure 3. Idealized M-unit profiles (solid) and lines of interpolation (dashed).

Two external implementation data variables applicable to both the TESS-NC operator and to the calling application designer are r_{max}, the maximum APM CSCI output range, and h_{max}, the maximum APM CSCI output height. These two parameters are required by the APM CSCI to determine the horizontal and vertical resolution, respectively, for internal range and height calculations based on the current values of n_{rout} and n_{zout}. Any value of r_{max} and h_{max} is allowed for the convenience of the TESS-NC operator and the calling application designer, provided $r_{max} \geq 5$ km, and $h_{max} \geq 100$ m. For example, the TESS-NC operator may desire a coverage diagram which extends to a range of 500 kilometers (km). In addition to accommodating the desires of the operator, specification of such a convenient maximum range eases the burden for the application designer in determining incremental tick marks for the horizontal axis of the display.

Provided the value of the parameter *lerr12* is set to '.false.', if the furthest environment profile range is less than r_{max}, the APM CSCI will automatically create an environment profile at r_{max} equal to the last profile specified, making the environment homogeneous from the range of the last profile specified to r_{max}. For example, a profile is input with an accompanying range of 450 km. If the TESS-NC operator chooses an r_{max} of 500 km, the APM CSCI will continue loss calculations to 500 km, keeping the refractivity environment homogeneous from 450 km to 500 km.

If *lerr12* is set to '.true.' and the furthest environment profile range is less than r_{max}, then an error will be returned in i_{error} from the APMINIT CSC. This is to allow the TESS-NC CSCI application designer greater flexibility in how environment data is handled.

7.3 TERRAIN PROFILE DATA ELEMENT

The terrain profile should consist of linear piece-wise segments specified in terms of range/height pairs. All range values must be increasing, and the first terrain height value must be at range zero. General ground composition types can be specified (table 4), along with corresponding ranges over which the ground type is to be applied. If ground type "User Defined" is specified ($igrnd_i = 7$), then

numeric values of relative permittivity and conductivity must be given. If horizontal antenna polarization is specified, the APM CSCI will assume perfect conductivity for the entire terrain profile and will ignore any information regarding ground composition. If vertical antenna polarization is specified, then information regarding ground composition must also be specified.

The maximum height, h_{max}, must always be greater than the minimum height, h_{min}. Also, a value of h_{max} must be given such that it is larger than the maximum elevation height along a specified terrain profile.

Provided *lerr6* is set to '.false.', if the furthest range point in the terrain profile is less than r_{max}; the APM CSCI will automatically create a height/range pair as part of the terrain profile at r_{max} with elevation height equal to the last height specified in the profile, making the terrain profile flat from the range of the last profile point specified to r_{max}. For example, a terrain profile is input where the last height/range pair is 50 m in height with an accompanying range of 95 km. If the TESS-NC operator chooses an r_{max} of 100 km, the APM CSCI will continue loss calculations to 100 km, keeping the terrain profile flat from 95 km to 100 km with an elevation height of 50 m.

If *lerr6* is set to '.true.' and the furthest range point is less than r_{max}, then an error will be returned in i_{error} from the APMINIT SU. This is to allow the TESS-NC CSCI application designer greater flexibility in how terrain data are handled.

7.4 ACRONYM AND ABBREVIATIONS

Table 102 is a glossary of acronyms and abbreviations used within this document.

Table 102. Acronyms and abbreviations.

Term	Definition
AMIN	Minimum of variables within parenthesis
AMAX	Maximum of variables within parenthesis
AP	Angle trace function
APM	Advanced Propagation Model
Centibel	One-hundredth of the logarithm of a quantity
COMMON BLOCK	Allows two or more FORTRAN Sus to share variables without having to pass them as arguments
COS	Cosine function
CMPLX	Data conversion to complex number
CSCI	Computer software configuration item
dB	Decibel
decibel	times the logarithm of a quantity
EM	electromagnetic
FE	Flat earth

Table 102. Acronyms and abbreviations. (Continued)

Term	Definition
FFT	Fast Fourier Transform
FORTRAN	Formula Translation
HP	Height trace function
IMAG	Imaginary part of complex number
INT	Integer value of
km	Kilometers
LOG	Logarithm to base 10
LN	Natural logarithm
m	Meters
M	Modified refractivity units
MHz	Megahertz
M-unit	Refractivity measurement unit
μV/m	Microvolts per meter
N/A	Not applicable
NINT	Round real number
PE	Parabolic Equation
PINT	Interpolation function
PLINT	Interpolation function
P space	Phase space
RADA1	Angle trace function
Radian	Unit of angular measurement
REAL	Real part of complex number
RO	Ray Optics
RP	Range trace function
SIGN	Sign transfer function
SIN	Sine function
SIN^{-1}	Inverse sine function
S/m	Conductivity unit Siemans per meter
Sin(X)/X	Sine(X)/X
SRS	Software Requirements Specification

Table 102. Acronyms and abbreviations. (Continued)

Term	Definition
SU	Software unit
TAN^{-1}	Inverse tangent function
TESS-NC	Tactical Environmental Support System-Next Century

7.5 SDD VARIABLE NAME, FORTRAN VARIABLE NAME CROSS REFERENCE

Table 103 is a cross reference of variable names used within the body of this document and the FORTRAN variable names as used within the APM CSCI source code of section 8, appendix A. Included are the SDD variable name, its description, the FORTRAN source code name, and the designation of the FORTRAN COMMON BLOCK name, if applicable.

Table 103. Variable name cross reference.

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
a	Complex coefficient of partial linear solution to homogeneous equation	ar	N/A
a	Angle defined by equ. 115 in EREPS 3.0 User's Manual NraD TD 2648, pp. 105	al	N/A
a_0	Angle at start of ray trace step	a0	N/A
a_1	Angle at end of ray trace step	a1	N/A
a_2	Tangent angle for receiver height $zout_j$	ang2	N/A
a_{atz}	Local ray or propagation angle at height z_{lim} and range r_{atz}	aatz	TRVAR
abs_{hum}	Absolute humidity near the surface	abshum	REFRACTIVITY
$ad1$	Array of tangent ranges from source height – used with terrain profile	ad1()	N/A
$adif$	Height differences between ant_{ref} and all output receiver heights	adif()	N/A
a_{ek}	$^4/_3$ effective earth's radius	aek	N/A
a_{ek2}	Twice $^4/_3$ effective earth's radius	aek2	MISCVAR
a_{fac}	Antenna pattern parameter (depends on I_{pat} and μ_{bw})	afac	PATTERN
a_{launch}	Launch angle used which, when traced, separates PE and XO regions from the RO region	alaunch	TRVAR

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
α	Source elevation angle	a	N/A
α_d	Direct-path ray angle	alphad	MISCVAR
a_{dif}	The difference between current and previous outgoing propagation angles	angdif	N/A
α_{ld}	LOG of antenna pattern factor for α_d where α_d represents lowest direct ray angle in optical region	ald	N/A
α_{lim}	Elevation angle of the RO limiting ray	alflim	N/A
α_{pat}	Elevation angle relative to the antenna elevation angle	udif	N/A
α_r	Reflected-path ray angle	alphar	N/A
α_u	Maximum tangent ray angle from the source to the terrain peak along profile height	angu	N/A
α_V	Surface impedance term	alphav	IMPEDANCE
α_{mlim}	Elevation angle of RO limiting ray in radians. Used to initialize launch angle in the GETTHMAX SU	amlim	N/A
ant_{ht}	Transmitting antenna height above local ground	antht	SYSTEMVAR
ant_{k_o}	Height-gain value at souce	antko	N/A
ant_{ref}	Transmitting antenna height relative to h_{minter}	antref	MISCVAR
a_{temp}	Temporary angle variable	atemp	N/A
$araftt$	Array of angles after smoothing operation	araft()	N/A
$arbef$	Array of angles before smoothing operation	arbef()	N/A
a_{start}	Elevation angle at start of ray step	aa	N/A
b	Complex coefficient of partial linear solution to homogeneous equation	br	N/A
b	Angle defined in equ. 116 in EREPS 3.0 User's Manual NRaD TD 2648, pp. 105	be	N/A
β	Terminal elevation angle	ab	N/A
β_d	Direct ray terminal elevation angle	betad	N/A
β_r	Reflected ray terminal elevation angle	betar	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
C_1	Coefficient used in vertical polarization calculations	c1	IMPEDANCE
C_2	Coefficient used in vertical polarization calculations	c2	IMPEDANCE
C_{1x}	Constant used to propagate C_1 by one range step	c1x	IMPEDANCE
C_{2x}	Constant used to propagate C_2 by one range step	c2x	IMPEDANCE
con	$10^{-6} k_o$	con	PE
c_n	Constant equals $\Delta p / k_o$	cnst	PE
ct_1	Quantity defined in equ. 124 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	ct1	N/A
ct_2	Quantity defined in equ. 125 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	ct2	N/A
$curang$	Array of current local angles for each ray being traced in XO region	curang()	N/A
$curht$	Array of current local heights for each ray being traced in XO region	curht()	N/A
$curng$	Array of current local ranges for each ray being traced in XO region	curng()	N/A
ΔFd_{lo}^2	Difference in direct ray magnitude along Δx_{RO} below desired APM output point	dfsdlo	N/A
ΔFd_{hi}^2	Difference in direct ray magnitude along Δx_{RO} above desired APM output point	dfsdhi	N/A
ΔFr_{lo}^2	Difference in reflected ray magnitude along Δx_{RO} below desired APM output point	dfsrlo	N/A
ΔFr_{hi}^2	Difference in reflected ray magnitude along Δx_{RO} above desired APM output point	dfsrhi	N/A
ΔH_o	Frequency gain function correction term defined in equ. 127 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	delho	N/A
$\Delta\Omega_{hi}$	Difference in total phase lag angle along Δx_{RO} above desired APM output point	danghi	N/A
$\Delta\Omega_{lo}$	Difference in total phase lag angle along Δx_{RO} below desired APM output point	danglo	N/A
Δp	Mesh size in angle- (or p-) space	delp	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
Δr_{out}	Output range step	drout	OUTRH
Δr_{PE}	PE range step	dr	PE
Δr_{PE2}	½ PE range step	dr2	PE
Δr_{temp}	Range step for ray tracing	drtemp	N/A
$\Delta\Theta$	Angle difference between mesh points in p-space	dtheta	N/A
Δx_{RO}	RO range interval	delxRO	RO
Δz_{out}	Output height increment	dzout	OUTRH
Δz_{PE}	PE mesh height increment (bin width in z-space)	delz	PE
Δz_{PE2}	2 Δz_{PE}	dz2	PE
d_1	Range from source to tangent point	d1	N/A
d_{1s}	Tangent range from the source for smooth surface	d1s	N/A
d_2	Range from receiver to tangent point	d2	N/A
$d2s$	Array of tangent ranges for all output receiver heights over smooth surface	d2s()	N/A
$d\alpha$	μ_{bwr} / 2	dalpha	N/A
$dielec$	Two-dimensional array containing the relative permittivity and conductivity, $dielec_{1,i}$ and $dielec_{2,i}$, respectively	dielec(,)	N/A
dum	Dummy array used for temporary storage	dum()	N/A
$dxd\alpha$	Derivative of range with respect to elevation angle	dxda	N/A
$dxd\alpha_d$	Derivative of range with respect to α_d	dxdad	N/A
$dxd\alpha_r$	Derivative of range with respect to α_r	dxdar	N/A
$dzd\alpha_d$	Derivative of height with respect to α_d	dzdad	N/A
$dzd\alpha_r$	Derivative of height with respect to α_r	dzdar	N/A
e_k	$^4/_3$ effective earth's radius factor	ek	N/A
$envpr$	Complex [refractivity] phase term array interpolated every Δz_{PE} in height	envpr()	N/A
ε_r	Relative permittivity	epsilon	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
η_s	Quantity defined in equ. 126 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	etas	N/A
F^2	Propagation factor squared	fsq	N/A
$f(\vartheta_1)$	Antenna pattern factor for angle, ϑ_1	factr	N/A
$f(\alpha_d)$	Antenna pattern factor for direct ray	facd	N/A
$f(-\alpha_r)$	Antenna pattern factor for reflected ray	facr	N/A
$farray$	Field array to be propagated one range step in free space	farray()	N/A
Fd^2	Magnitude array, direct ray	dmagsq(,)	N/A
F_{fac}	Propagation factor in dB	ffacdb	N/A
F_{fac}	Propagation factor in dB	ffac	N/A
$ffacz$	Array containing propagation factor, range, and propagation angle at z_{lim}	ffacz()	N/A
$ffrout$	Array of propagation factors at each output range beyond r_{atz} and at height, z_{lim}	ffrout()	N/A
$filt$	Cosine-tapered (Tukey) filter array	filt()	N/A
$filtp$	Array filter for spectral estimation calculations	filtp()	N/A
f_{MHz}	Frequency	freq	SYSTEMVAR
f_{norm}	Normalization factor	fnorm	PE
f_r	Fractional bin used for interpolation	fr	N/A
fr^2	Magnitude array, reflected ray	rmagsq(,)	RO
f_{rac}	Fractional distance between pl_1 and pl_2	frac	N/A
$frac_{RO}$	RO range interval fraction (0.0 to 0.25)	fracRO	N/A
$frsp$	Complex free space propagator term array	frsp()	N/A
$fslr$	Free-space loss array for output ranges	fslr()	N/A
fsl_{rout}	Free-space loss at range r_{out}	fslrout	N/A
f^2_{sum}	Square of coherent sum of directand reflected rays	ffac2	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
f_{ter}	Logical flag indicating if terrain profile has been specified: .true. = Terrain profile specified .false. = Terrain profile not specified	fter	MISCVAR
fv	Fraction range for profile interpolation	fv	N/A
γ_a	Surface specific attenuation	gammaa	REFRACTIVITY
γ_o	Oxygen absorption	gammao	N/A
γ_w	Water absorption	gammaw	N/A
gas_{att}	Gaseous absorption attenuation rate	gasatt	ABSORB
gr	Intermediate M-unit gradient array, RO region	gr()	N/A
$grad$	Two-dimensional array containing gradients of each profile used in XO calculations	grad(,)	N/A
g_{rd}	Refractivity gradient	grd	N/A
h_0	Height at start of ray trace step	h0	N/A
h_1	Height at end of ray trace step	h1	N/A
H_1	Quantity defined in equ. 120 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	hor1	N/A
H_2	Quantity defined in equ. 121 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	hor2	N/A
$hfang$	Cut-back angles in degrees	hfang()	N/A
$hfangr$	Cut-back angles in radians	hfangr()	N/A
$hffac$	Cut-back antenna pattern factors	hffac()	N/A
h_{large}	Maximum height limit for last level in height/refractivity profiles	hlarge	N/A
$hlim$	Array containing height at each output range separating the RO region from the PE (at close ranges) and XO (at far ranges) regions	hlim()	N/A
h_{max}	Maximum output height with respect to mean sea level	hmax	INPUTVAR
h_{min}	Minimum output height with respect to mean sea level	hmin	INPUTVAR
h_{minter}	Minimum height of terrain profile	hminter	REFPROF

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
hm_{ref}	Height relative to h_{minter}	Hmref	MISCVAR
$hmsl$	Two-dimensional array containing heights with respect to mean sea level of each profile. Array format must be $hmsl_{i,j}$ = height of i^{th} level of j^{th} profile; $j = 1$ for range-independent cases	hmsl(,)	N/A
h_o	Effective scattering height—defined in equ. 109 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 105	h0	N/A
H_o	Frequency gain function defined in equ. 119 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	Bigh	N/A
$href$	Heights of refractivity profile with respect to y_{ref}	href()	N/A
ht	PE mesh height array of size, n_{fft}	ht()	N/A
$htdum$	Height array for current interpolated profile	htdum()	N/A
$htemp$	Heights at which ray is traced to every range in *rtemp*	htemp()	TRVAR
h_{termax}	Maximum terrain height along profile path	htermax	N/A
h_{test}	Minimum height at which all trapping refractivity features are below	htest	N/A
$htfe$	Array containing the height at each output range separating the FE region from the RO region (full hybrid mode), or the FE region from the PE region (partial hybrid mode)	htfe()	N/A
h_{thick}	Thickness of highest trapping layer from all refractivity profiles	hthick	N/A
ht_{lim}	Maximum height relative to h_{minter}	htlim	MISCVAR
$htout$	Final height for each ray traced in XO region at range, r_{out}	htout()	N/A
htr	Two-dimensional array containing heights of each profile used in XO calculations	htr()	N/A
h_{trap}	Height of highest trapping layer from all refractivity profiles	htrap	N/A
ht_{ydif}	$ht_{lim} - y_{fref}$	htydif	RO

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
i_{ap}	Index indicating when the local ray angle becomes positive in array *raya*	iap	TRVAR
i_{av}	Number of points over which to perform average smoothing	iav	N/A
i_{bmsl}	Integer flag indicating if y_{ref} is below mean sea level (msl) $i_{bmsl} = 0$: y_{ref} not below msl $i_{bmsl} = 1$: y_{ref} below msl	ibmsl	N/A
i_{error}	Error flag	ierror	N/A
i_{extra}	Extrapolation flag for refractivity profiles entered below mean sea level $i_{extra} = 0$; extrapolate to minimum terrain height standard atmosphere gradient $i_{extra} = 1$; extrapolate to minimum terrain height using first gradient in profile	iextra	REFRACTIVITY
i_{flag}	Integer flag indicating height at which to reference the refractivity profile $i_{flag} = 0$; adjust profile relative to h_{minter} $i_{flag} = 1$; adjust profile relative to local ground height above h_{minter}	iflag	N/A
i_g	Counter indicating current ground type being modeled	ig	IMPEDANCE
i_{gr}	Number of different ground types specified	igr	TERRAIN
i_{grad}	Index of current gradient level in *grad*	igrad	N/A
igrd	Integer indexes indicating at what refractive gradient level to begin ray tracing for next XO range step for each ray in XO region	igrd()	N/A
igrnd	Integer array containing ground type composition for given terrain profile—can vary with range. Different ground types are: 0 = Seawater 1 = Freshwater 2 = Wet ground 3 = Medium dry ground 4 = Very dry ground 5 = Ice at -1 degree C 6 = Ice at -10 degree C 7 = User-defined (in which case, values of relative permittivity and conductivity must be given)	igrnd()	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
i_{hybrid}	Integer indicating which submodels will be used: 0 = Pure PE model 1 = Full hybrid model (PE + FE + RO + XO) 2 = Partial hybrid model (PE + XO)	ihybrid	MISCVAR
i_{p1}	First output height point index in *zout* where propagation loss will be computed at previous PE range	ip1	N/A
i_{p2}	First output height point index in *zout* where propagation loss will be computed at current PE range	ip2	N/A
i_{pat}	Antenna pattern type i_{pat} = 1 : Omni-directional i_{pat} = 2 : Gaussian i_{pat} = 3 : Sine(x)/x i_{pat} = 4 : Cosecant-squared i_{pat} = 5 : Generic height-finder i_{pat} = 6 : User-defined height-finder	ipat	SYSTEMVAR
i_{peak}	Bin # in *spectr* corresponding to the peak magnitude	ipeak	N/A
i_{pol}	Polarization flag: 0 = Horizontal polarization 1 = Vertical polarization	ipol	SYSTEMVAR
i_{quit}	Integer flag indicating to quit tracing current ray and begin again with a new launch angle	iquit	N/A
i_{ratz}	Index of output range step in which to begin storing propagation factor and outgoing angle for XO region	iratz	TRVAR
i_{ROn}	Array index for next range in RO region	iROn	RO
i_{ROp}	Array index for previous range in RO region	iROp	RO
i_{rp}	Counter for current refractivity/gradient profile being used from *grad*	irp	N/A
i_{rps}	Starting index counter for refractivity profiles	irps	N/A
i_{rtemp}	Temporary number of range steps (used for ray tracing)	irtemp	N/A
i_s	Counter for current profile	is	REFPROF
i_{start}	Array index for height in RO region corresponding to ant_{ref}	istart	RO

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
i_{stp}	Current output range step index	istp	N/A
i_{tp}	Number of height/range points in profile	itp	TERRAIN
i_{tpa}	Number of height/range points pairs in profile *tx, ty*	itpa	MISCVAR
i_{tropo}	Troposcatter calculation flag: i_{tropo} = 0; no troposcatter calcs i_{tropo} = 1; troposcatter calcs	itropo	INPUTVAR
i_{type}	Ray type (direct or reflected) flag	itype	N/A
ix	Height counter index	ix	N/A
i_{xo}	Number of range steps in XO calculation region	ixo	MISCVAR
i_{xostp}	Current output range step index for XO calculations	ixostp	N/A
iz	Number of propagation factor, range, and angle triplets stored in *ffacz*	iz	XO
iz_e	Ending index in *curang*, *curng*, and *curht* to trace to r_{out}	ize	N/A
i_{zg}	Number of output height points corresponding to local ground height at current output range, r_{out}	izg	MISCVAR
iz_{inc}	Integer increment for storing points at top of PE region (i.e., points are stored at every iz_{inc} range step)	izinc	XO
iz_{max}	Maximum number of points allocated for arrays associated with XO calculations	izmax	XO
iz_s	Starting index in *curang*, *curng*, and *curht* to trace to r_{out}	izs	N/A
j_e	Ending receiver height index at which to compute troposcatter loss	je	N/A
j_{end}	Index at which valid loss values in *mloss* end	jend	N/A
j_{fe}	Ending index within *mloss* of FE loss values	jfe	N/A
j_{fs}	Starting index within *mloss* of FE loss values	jfs	N/A
j_{max}	Array index for maximum output height in RO region	jmax	N/A
j_{min}	Array index for minimum output height in RO region	jmin	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
j_{re}	Ending index within *mloss* of RO loss values	jre	N/A
j_{rs}	Starting index within *mloss* of RO loss values	jrs	N/A
j_s	Starting receiver height index at which to compute troposcatter loss	js	N/A
j_{start}	Index at which valid loss values in *mloss* start	jstart	N/A
j_{t1}	Index counter for $ad1$ and $\vartheta 1t$ arrays	jt1	TROPOV
j_{t2}	Index counter for tx and ty arrays indicating location of receiver range	jt2	TROPOV
j_{xe}	Index at which valid loss values in *mloss* end	jxe	N/A
j_{xs}	Index at which valid loss values in *mloss* start	jxs	N/A
j_{xstart}	Starting index within *mloss* of XO loss values	jxstart	N/A
jz_{lim}	PE bin # corresponding to z_{lim}, i.e., $z_{lim} = jz_{lim} \Delta z_{PE}$	jzlim	XO
k	Integer bin # in field U corresponding to height, z_r	nb	N/A
k	Grid point counter used in RO calculations	k	N/A
k_{abs}	Gaseous absorption calculation flag: $k_{abs} = 0$; no absorption loss $k_{abs} = 1$; compute absorption loss based on air temperature t_{air} and absolute humidity abs_{hum} $k_{abs} = 2$; compute absorption loss based on specified absorption attenuation rate γ_a	kabs	ABSORB
k_{bin}	Number of bins complex PE field is to be shifted	kbin	N/A
k_{hi}	k index above desired point	khi	N/A
k_{lo}	k index below desired point	klo	N/A
kmax	Array index for maximum angle in RO region at range, x_{Ron}	kmax	RO
kminn	Array index for minimum angle in RO region at range, x_{Ron}	kminn	RO
kminp	Array index for minimum angle in RO region at range, x_{Rop}	kminp	RO
k_o	Free-space wavenumber	fko	MISCVAR
k_t	Counter index for terrain profile arrays tx and ty	kt	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
k_{temp}	Temporary k_{lo} value	klotmp	N/A
ktr_l	Number of tangent ranges from source height	ktr1	TROPOV
l_{abscb}	Gaseous absorption loss	labscb	N/A
λ	Wavelength	wl	MISCVAR
lerr6	User-provided error flag that will trap on certain errors if set to '.true.'	lerr6	ERRORFLAG
lerr12	User-provided error flag that will trap on certain errors if set to '.true.'	lerr12	ERRORFLAG
levels	Number of levels in *gr*, *q* and *zrt* arrays	levels	RO
l_{new}	Temporary refractivity level counter	newl	N/A
ln_{fft}	Power of 2 transform size (i.e., $n_{fft} = 2^{ln_{fft}}$)	ln	PE
ln_{min}	Minimum power of 2 transform size	lnmin	PE
ln_p	Power of 2 transform size used in spectral estimation calculations (i.e., $n_p = 2^{lnp}$)	lnp	SPEC
lvl	Number of height levels in each profile used in XO calculations	lvl()	N/A
lvlep	Number of height/refractivity levels in profile *refdum* and *htdum*	lvlep	REFPROF
lvlp	Number of height/refractivity levels in profiles	lvlp	REFRACTIVITY
mloss	Propagation loss array	mloss	N/A
μ_o	Antenna elevation angle in degrees	elev	SYSTEMVAR
μ_{or}	Antenna pattern elevation angle in radians	elv	PATTERN
μ_{bw}	Antenna vertical beamwidth in degrees	bwidth	SYSTEMVAR
μ_{bwr}	Antenna vertical beamwidth in radians	bw	PATTERN
μ_{max}	Limiting angle for Sin(X)/X and generic height finder antenna pattern factors	umax	PATTERN
$n_{3/4}$	¾ n_{fft}	n34	PE
n_4	¼ n_{fft}	n4	PE
nc^2	Array of complex dielectric constants	cn2()	N/A
n_{fft}	Transform size	n	PE

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
n_{facs}	Number of user-defined cut-back angles and cut-back pattern factors	nfacs	SYSTEMVAR
$nlvl$	Number of levels in new profile	nlvl	REFPROF
n_{m1}	$n_{fft} - 1$	nm1	PE
n_{p34}	$\frac{3}{4}\, n_p$	np34	SPEC
n_{p4}	$\frac{1}{4}\, n_p$	np4	SPEC
n_p	Number of bins in upper PE region to consider for spectral estimation	npnts	SPEC
n_{prof}	Number of refractivity profiles	nprof	REFRACTIVITY
n_{rout}	Integer number of output range points desired	nrout	INPUTVAR
n_s	Transform size for spectral estimation calculations	ns	SPEC
n_{xo}	Number of rays traced (i.e., height points, in XO region)	nxo	N/A
n_{zout}	Integer number of output height points desired	nzout	INPUTVAR
Ω	Total phase angle	phdif	N/A
Ω	Total phase angle array	omega(,)	RO
p_1	Refractivity variable	p1	N/A
p_2	Refractivity variable	p2	N/A
p_{elev}	Sine of antenna elevation angle	pelev	PATTERN
$pfac_{min}$	Minimum propagation factor	pfacmin	N/A
pf_{db}	Propagation factor in dB at current PE range, r, at height, z_{lim}	pfdb	N/A
pf_{dblst}	Propagation factor in dB at previous PE range, r_{last}, at height, z_{lim}	pfdblst	N/A
$pf_{r_{atz}}$	Propagation factor in dB at range, r_{atz}, and height, z_{lim}	pfratz	N/A
φ	Phase lag angle of reflected ray	rphase	N/A
pl_1	Path loss variable	pl1	N/A
pl_2	Path loss variable	pl2	N/A
pl_{cnst}	Constant used in determining propagation loss (pl_{cnst} = 20 LOG(2 k_o))	plcnst	MISCVAR

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
pl_d	Path length from range x_r	pld	N/A
pl_d	Path length difference from x for direct ray	pld	N/A
pl_r	Path length difference from x for reflected ray	plr	N/A
p_{mag}	Interpolated magnitude of complex PE field	pmag	N/A
$prfac$	Propagation factor for each ray traced in XO region range r_{out}	prfac()	N/A
$profint$	Profile interpolated to every Δz_{PE} in height	profint()	N/A
ψ	Grazing angle	angle	N/A
ψ	Grazing angle	psi	N/A
ψ_{lim}	Grazing angle of limiting ray	psilim	RO
q	Intermediate M-unit difference array, RO region	q()	N/A
q_t	Quantity defined in equ. 128 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 107	qt	N/A
r	Current PE range	r	N/A
R	Coefficient used in C_1 and C_2 calculations.	rk	IMPEDANCE
r_0	Range at start of ray trace step	r0	N/A
r_1	Path length for direct-ray path	r1	N/A
r_1	Quantity defined in equ. 122 in EREPS 3.0 User's Manual, NRaD TD 2648, pp. 106	r1	N/A
r_1	Range at end of ray trace step	r1	N/A
r_2	Path length for reflected-ray path	r2	N/A
r_2	Quantity defined in equ. 123 in EREPS 3.0 User's Manual, NraD TD 2648, pp. 106	r2	N/A
rad	Radical for square root test in ray trace step	rad	N/A
r_{adc}	Radians to degrees conversion factor	radc	N/A
r_{ange}	Range for profile interpolation	range	N/A
$ratio$	Fractional range term used for interpolation	ratiox	N/A
$ratio_k$	Fraction of one k index (0. To 1.)	ratiok	N/A
r_{atz}	Range at which z_{lim} is reached (used for hybrid model)	ratz	TRVAR

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
raya	Array containing all local angles of traced ray a_{launch} at each i_{rtemp} range	raya()	N/A
$rdif_1$	Range difference between adjacent terrain points	rdif1	N/A
$rdif_2$	Range difference between adjacent terrain points	rdif2	N/A
r_{difsum}	Sum of adjacent terrain point differences	rdifsum	N/A
rdt	Array of minimum ranges at which diffraction field solutions are applicable (for smooth surface) for all output receiver heights	rdt()	N/A
refdum	M-unit array for current interpolated profile	refdum()	N/A
refmsl	Two-dimensional array containing refractivity with respect to mean sea level of each profile. Array format must be $refmsl_{i,j}$ = M-unit at i^{th} level of j^{th} profile; $j = 1$ for range-independent cases	refmsl(,)	N/A
refref	Refractivity profile with respect to y_{ref}	refref()	N/A
r_f	Constant used for troposcatter calculations	rf	TROPOV
rfac1	Propagation factor at valid output height points from PE field at range, r_{last}	rfac1()	N/A
rfac2	Propagation factor at valid output height points from PE field at range, r	rfac2()	N/A
r_{fix}	Fixed range increment of terrain profile	rfix	N/A
r_{flat}	Maximum range at which the terrain profile remains flat from the source	rflat	N/A
r_{frac}	Ratio between adjacent terrain point differences	rfrac	N/A
rgrnd	Array containing ranges at which varying ground types apply	rgrnd()	N/A
r_{hor1}	Minimum range at which diffraction field solutions are applicable—determined for 0 receiver height	rdhor1	N/A
r_{last}	Previous PE range	rlast	N/A
r_{log}	10 LOG(PE range, r)	rlog	MISCVAR
rlogo	Array containing 20 times the logarithm of all output ranges	rlogo()	N/A
r_{loglst}	10 LOG(previous PE range, r_{last})	rloglst	MISCVAR

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
rloss	Propagation loss	rloss()	N/A
rm	Intermediate M-unit array, RO region	rm()	N/A
r_{mag}	Magnitude of reflection coefficient	rmag	N/A
r_{max}	Maximum specified range	rmax	INPUTVAR
r_{mid}	Range at which interpolation for range-dependent profiles is performed	rmid	N/A
r_{mmax}	Maximum M-unit value of refractivity profile at range 0	rmmax	N/A
r_{mmin}	Minimum M-unit value of refractivity profile at range 0	rmmin	N/A
R_{ng}	Complex refractive index	rng	N/A
rngout	Array containing all desired output ranges	rngout()	N/A
rngprof	Ranges of each profile. $rngprof_i$ = range of i^{th} profile	rngprof()	N/A
r_o	Current ending range for ray trace step	ro	N/A
root	Array of R_T to the i^{th} power (e.g., $root_i = R_T^i$)	root()	N/A
rootm	Array of $-R_T$ to the i^{th} power (e.g., $rootm_i = (-R_T)^i$)	rootm()	N/A
r_{out}	Current output range	rout	N/A
r_{pest}	Range at which PE loss values will start being calculated	rpest	MISCVAR
r_{sq}	Square of current output range	rsq	N/A
rsqrd	Array containing the square of all desired output ranges	rsqrd()	N/A
R_T	Complex root of quadratic equation for mixed transform method based on Kuttler's formulation	rt	IMPEDANCE
rt_1	$r_f * ant_{ref}$	r1t	TROPOV
rtemp	Range steps for tracing to determine maximum PE angle	rtemp()	TRVAR
r_{tst}	Range at which to begin RO calculations (equal to 2.5 km)	rtst	N/A
$R_{V,H}$	Complex reflection coefficient for vertical (V) and horizontal (H) polarization	refcoef	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
rv_1	Range of the previous refractivity profile	rv1	N/A
rv_2	Range of the next refractivity profile	rv2	REFPROF
σ	Conductivity	sigma	N/A
s	Quantity defined equ. 110 in EREPS 3.0 User's Manual NRaD TD 2648, pp. 105	s	N/A
s_{bw}	Sine of antenna vertical beam width	sbw	PATTERN
s_{gain}	Normalization factor used in starter field calculation	sgain	N/A
slp	Slope of each segment of terrain	slp()	N/A
sn_1	Term used in troposcatter loss calculation	sn1	TROPOV
sn_{ref}	Surface refractivity	snref	N/A
$spectr$	Spectral amplitude of field	spectr()	N/A
sum_1	Summation term in determining a	sum1	N/A
sum_2	Summation term in determining b	sum2	N/A
ta_{ek}	Twice the effective earth's radius	twoka	MISCVAR
$\vartheta 0$	Array of angles used to determine common volume scattering angle	theta0()	N/A
ϑ_1	Tangent angle from source height	theta1	N/A
ϑ_2	Tangent angle from receiver height	theta2	N/A
ϑ_{1s}	Tangent angle from source (for smooth surface)	theta1s	TROPOV
$\vartheta 2s$	Array of tangent angles from all output receiver heights—used with smooth surface	theta2s()	N/A
$\vartheta 1t$	Array of tangent angles from source height - used with terrain profile	th1()	N/A
Θ_{max}	Maximum propagation angle in PE calculations	thetamax	N/A
Θ_{75}	75% of maximum propagation angle in PE calculations	theta75	PE
ϑ_{out}	Outgoing propagation angle determined at top of PE region	thout	N/A
t_{air}	Air temperature near the surface	tair	REFRACTIVITY
$terx$	Range points of terrain profile	terx()	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
$tery$	Height points of terrain profile	tery()	N/A
t_{loss}	Troposcatter loss in dB	tloss	N/A
$tlst$	Troposcatter loss term	tlst	N/A
$tlst_s$	Troposcatter loss term for smooth surface case	tlsts	TROPOV
tol	Height tolerance	tol	N/A
t_{st}	Current largest tangent angle from source	tst	N/A
tx	Range points of terrain profile	tx()	N/A
ty	Adjusted height points of terrain profile	ty()	N/A
U	Complex field at current PE range, r	u()	N/A
$Ulast$	Complex field at previous PE range, r_{last}	ulst()	N/A
w	Difference equation of complex PE field	w()	N/A
x	Current output range	x	N/A
x	Field array to be transformed—dimensioned $2^{n_{fft}}$ in calling SU	x()	N/A
$xdum$	Real part of complex field array	xdum()	N/A
xo_{con}	Constant used in determining ϑ_{out}	xocon	SPEC
xp	Real part of spectral portion of PE field	xp()	N/A
x_r	Terminal range—called x_{ROn} in ROCALC SU	rout	N/A
$x_{reflect}$	Range at which ray is reflected	xreflect	RO
x_{ROn}	Next range in RO region	xROn	RO
x_{ROp}	Previous range in RO region	xROp	RO
x_{temp}	Temporary range in ray trace step	xtemp	N/A
x_{sum}	Running sum of range during ray trace	xsum	N/A
xx	Fractional range for interpolation	xx	N/A
y_{ch}	Height of terrain at the current PE range relative to hm_{ref}	ych	N/A
y_{cur}	Height of ground at current range, r	ycur	MISCVAR
y_{curm}	Height of ground midway between last and current PE range	ycurm	MISCVAR

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
y_{diff}	y_{cur}- y_{last}	ydiff	N/A
$ydum$	Imaginary part of complex field array	ydum()	N/A
y_{fref}	Ground elevation height at source	yfref	MISCVAR
y_{last}	Height of ground at previous range, r_{last}	ylast	MISCVAR
y_{lh}	Height of terrain at the previous PE range relative to hm_{ref}	ylh	N/A
ym	Particular solution of difference equation	ym()	N/A
y_n	Height of terrain at current range for traced ray	yn	N/A
ym	Particular solution of difference equation	ym()	N/A
y_n	Height of terrain at current range for traced ray	yn	N/A
y_{nt}	Height of terrain at source	ynt	N/A
yp	Imaginary part of spectral portion field	yp()	N/A
y_{ref}	Ground elevation height at current range	yref	N/A
z_d	Terminal height of direct ray	zd	N/A
z_{int}	Interpolated terrain elevation at current output range	zint	N/A
z_k	Height of k^{th} RO index	zk	N/A
z_{lim}	Height limit for PE calculation region	zlim	PE
z_{limt}	ht_{lim}-10^{-3}	zlimt	N/A
z_m	Output receiver height relative to "real" antenna height and adjusted for earth curvature	zm	N/A
z_{max}	Total height of the FFT/PE calculation domain	zmax	PE
$zout$	Array containing all desired output heights referenced to h_{minter}	zout()	N/A
$zoutma$	Array output heights relative to "real" ant_{ref}	zoutma()	N/A
$zoutpa$	Array output heights relative to "image" ant_{ref}	zoutpa()	N/A
z_p	Output receiver height relative to "image" antenna height and adjusted for earth curvature	zp	N/A
z_r	Receiver height	height	N/A
z_r	Terminal height of reflected ray	zr	N/A
zRO	Array of output heights in RO region	zro()	N/A

Table 103. Variable name cross reference. (Continued)

SDD Variable Name	Description	FORTRAN Source Code Name	FORTRAN Common Block Name
zrt	Intermediate height array, RO region	zrt()	N/A
z_{test}	Height in PE region that must be reached for hybrid model	ztest	N/A
z_{tol}	Height tolerance for Newton's method	ztol	RO

APPENDIX A
FORTRAN SOURCE CODE FOR APM CSCI

A.1 SUBROUTINE APMINIT

```
! Version 1.0
! Author: Amalia E. Barrios
!         SPAWARSYSCEN SAN DIEGO D883
!         49170 Propagation Path
!         San Diego, CA  92152-7385
!         e-mail: barrios@spawar.navy.mil
!         phone: (619) 553-1429
!         fax: (619) 553-1417

! Summary: These routines model tropospheric radiowave propagation over
!          variable terrain and calculates propagation loss vs. height and
!          range. Propagation loss is displayed in dB contours on a height vs.
!          range plot. APM is based on the Radio Physical Optics (RPO) model
!          developed by Herb Hitney (SPAWARSYSCEN SAN DIEGO) and the Terrain
!          Parabolic Equation Model (TPEM) developed by Amalia Barrios
!          (SPAWARSYSCEN SAN DIEGO). The parabolic equation sub-model is based
!          on the split-step Fourier PE method and was originally developed
!          from an early PE model called PEPC, written by Fred Tappert.

!          Propagation loss over variable terrain is modeled by shifting
!          the field an appropriate number of bin widths correspond-
!          ing to the height of the ground.  The field is determined using the
!          smooth earth PE method. A hybrid capability is also included for
!          limited cases (low antenna heights and/or initial flat terrain).
!          The hybrid model consists of a flat earth (FE) region at very high
!          angles, a ray-optics (RO) model at intermediate angles, and
!          the split-step PE model below the lowest RO angle. An extended
!          optics model (XO) is used at heights above the PE region
!          and at ranges beyond the RO region.

! ***************************************************************************

! Variables in small letters in parameter lists are variables that are input
! or passed to called subroutines.  Variables in CAPS in parameter lists are
! returned from the called subroutines.

! ************************** SUBROUTINE APMINIT ******************************

! Module Name: APMINIT

! Module Security Classification: UNCLASSIFIED

! Purpose:  Initializes all variables used in APM subroutines for FE, RO,
!           and PE calculations. After initial units conversions have been
!           done, all calculations are in metric units.  Height and range
!           values are in meters and angles are in radians.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: ABSHUM, ANTHT, BWIDTH, ELEV, FREQ, GAMMAA, HFANG(),
!           HFFAC(), HMAX, HMIN, IGR, IPAT, IPOL, ITP,
!           ITROPO, LERR6, LERR12, LVLP, NPROF, NROUT, NZOUT,
!           RMAX, TAIR
!   Public: DIELEC(,), HMSL(,), IGRND(), REFMSL(,), RGRND(), RNGPROF(),
!           TERX(), TERY()
!   Parameters: PI
```

```
!   Data: AEK, EK, RADC, RTST

! OUTPUTS:
!   Argument List: IXOSTP, IERROR
!   Common: Most variables in common blocks ABSORB, IMPEDANCE, MISCVAR,
!           OUTRH, PATTERN, PE, REFPROF, RO, SPEC, TROPOV, TRVAR, XO
!   Public: All public arrays.

! Modules Used: APM_MOD

! Calling routines: MAIN DRIVER PROGRAM

! Routines called:
!   APM Specific: ALLARRAY_APM, ALLARRAY_PE, ALLARRAY_XO, DIEINIT, FFT, FILLHT,
!                 GASABS, GETALN, GETMODE, GETTHMAX, INTPROF, PHASE1,
!                 PHASE2, REFINIT, TERINIT, TROPOINIT, XYINIT
!   Intrinsic: ABS, ALOG10, AMAX1, AMIN1, ASIN, ATAN, COS, FLOAT, INT,
!              NINT, SIGN, SIN, SQRT

! GLOSSARY: See universal glossary for common variables, data variables
!           public arrays, and parameters.

!   Input Variables: NONE

!   Output Variables:
!      IXOSTP = Index of output range step at which XO model is to be applied.
!      IERROR = Integer value that is returned if any errors exist in input
!               data:
!               -6 : Last range in terrain profile is less than RMAX.
!                    (Will only return this error if error flag LERR6
!                    is set to .TRUE.).
!               -7 : Specified cut-back angles are not increasing. This is
!                    only tested for user-defined height-finder antenna
!                    pattern.
!               -8 : HMAX is less than maximum height of terrain profile.
!               -9 : Antenna height w.r.t. msl is greater than maximum
!                    height HMAX.
!              -10 : Beamwidth is less than or equal to zero for directional
!                    antenna pattern.
!              -12 : Range of last refractivity profile entered (for range
!                    dependent case) is less than RMAX. (This is returned
!                    from subroutine REFINIT).  Will only return this error
!                    if error flag LERR12 is set to .TRUE.).
!              -13 : Height of first level in any user-specified refrac-
!                    tivity profile is greater than 0.  First height must
!                    be at m.s.l. (0.) or <0. if below m.s.l.
!              -14 : Last gradient in any refractivity profile entered is
!                    negative. (This is returned from REFINIT).
!              -17 : Range points of terrain profile are not increasing.
!              -18 : First range point is not 0.
!              -42 : Minimum height input by user (HMIN) is greater then
!                    maximum height (HMAX).

!   Local Variables:
!       ALFLIM = Elevation angle of RO limiting ray in radians. Used to
!                initialize launch angle in GETTHMAX routine.
!       ANGU = Maximum tangent ray angle from source to terrain peak
!              along profile path.
!       ATEST = Tangent angle used for automatic calculation of maximum
!               propagation angle.  Only used for modes IHYBRID = 0, 2.
!       HMX = Maximum height to use when computing a maximum propagation
!             angle for PE calculations.  Only used for modes IHYBRID=0,2.
!       HTERMAX = Maximum terrain height along profile path in meters.
!       HTEST = Minimum height in meters at which all trapping
!               refractivity features are below (includes some slop).
!       HTHICK = Thickness in meters of highest trapping layer from all
!                refractivity profiles.
```

```
!       HTRAP = Height of highest trapping layer in meters from all
!               refractivity profiles.
!       RFIX = If terrain profile points are equally spaced, this is
!              automatically determined and range spacing is set to RFIX,
!              otherwise, RFIX = 0.
!       RFLAT = Maximum range in meters at which the terrain profile
!               remains flat from the source.
!       RKM = Maximum range in km.
!       RMMAX = Maximum M-unit value (x10e-6) of refractivity profile at
!               range 0.
!       RMMIN = Minimum M-unit value (x10e-6) of refractivity profile at
!               range 0.
!       THETAMAX = Maximum propagation angle used in the PE model.
!       ZTEST = This is the minimum height at which the PE model must
!               reach in order to contain all necessary refractivity and/or
!               terrain features.

subroutine apminit( IXOSTP, IERROR )

use apm_mod

complex clc, c2c

data c0 / 299.79245 /           !speed of light x 1e-6 m/s
data sdeg10 / .173648177 /      ! Sine of 10 degrees
data sdeg15 / .258819045 /      ! Sine of 15 degrees

ierror = 0
aek2 = 2.*aek
thetamax = 0.
kabs = 0
rpest = 0.

! Initialize flags for absorption calculations.

if(( tair .ne. 0. ) .or. ( abshum .ne. 0. )) kabs = 1
if( gammaa .ne. 0. ) kabs = 2

! Put lower limit on HMAX and RMAX

rmax = amax1( rmax, 5000. ) !Set max. range to no less than 5 km.
hmax = amax1( hmax, 100. )  !Set max. height to no less than 100 m.
if( hmin .ge. hmax ) then
   ierror = -42
   return
end if
hmin = amin1( hmin, hmax-100. )

dzout = (hmax-hmin) / float( nzout )
drout = rmax / float( nrout )

WL = c0 / freq
FKo = 2. * pi / WL
con = 1.e-6 * fko
fko2 = 2. * fko

!Loss term - add to 20log(r) to get free space loss.

plcnst=20.*alog10(fko2)

itpa = itp + 1

! Allocate and initialize all arrays associated with # of output height
! and range points.

call allarray_apm( IERROR )
if( ierror .ne. 0 ) return
```

```
do i = 1, nrout
   r = float(i) * drout
   rsqrd(i) = r * r
   rlogo(i) = 20. * alog10( r )
   fslr(i) = rlogo(i) + plcnst
   rngout(i) = r
end do

!  Calculate constants used to determine antenna pattern factor
! IPAT = 1 -> omni
! IPAT = 2 -> gaussian
! IPAT = 3 -> sinc x
! IPAT = 4 -> csc**2 x
! IPAT = 5 -> generic height-finder
! IPAT = 6 -> user-defined height-finder

if( nfacs .gt. 0 ) then
   hfangr = hfang * radc
   do i = 1, nfacs-1
      if( hfangr(i+1) .lt. hfangr(i) ) ierror = -7
   end do
   if( ierror .ne. 0 ) return
end if

if(( ipat .gt. 1 ) .and. ( bwidth .le. 1.e-4 )) ierror = -10
if( ierror .ne. 0 ) return

if( ipat .eq. 1 ) bwidth = 45. !For RO calculations.

bw = bwidth * radc
elv = elev * radc
bw2 = .5 * bw
if( ipat .eq. 2 ) then        !Gaussian
   afac = .34657359 / (sin( bw2 ))**2
   pelev = sin( elv )
elseif( ipat .eq. 4 ) then  !CSC**2
   sbw = sin( bw )
elseif( ipat .ne. 1 ) then
   afac = 1.39157 / sin( bw2 )
   a = pi / afac
   umax = atan( a / sqrt(1. - a*a)   )
end if

! Initialize terrain information.

call terinit( ANGU, RFIX, HTERMAX, IERROR )
if( ierror .ne. 0 ) return

! Setup output height arrays with respect to HMINTER.

yfref = 0.
if( fter ) yfref = ty(1)

do i = 0, nzout
   z = hmref + float(i) * dzout
   zout(i) = z
   zro(i) = z - yfref
   zoutma(i) = z - antref
   zoutpa(i) = z - yfref + antht
end do

! Determine what hybrid model(s) to use.

call getmode( RFLAT )

! Initialize refractivity arrays.
```

```
call refinit( HTRAP, HTHICK, RMMIN, RMMAX, IERROR )
if( ierror .ne. 0 ) return

! Initialize troposcatter variables.

if( itropo .eq. 1 ) call tropoinit

! Compute grazing angle limit based on 2.5 times Reed & Russell
! (p. 140) limit, but not less than .002 rad.  Double this value if
! more than one profile was entered, then adjust for trapping
! effects. Compute corresponding RO elevation angle limit at
! transmitter, ALFLIM.

psilim = amax1( .002, .04443 / (freq ** .3333333) )
IF (nprof .GT. 1) psilim = 2. * psilim
psilim = psilim + SQRT(ABS(2. * (rmmax - rmmin)))
alflim = SQRT(ABS(psilim ** 2 + 2. * (rm(istart) - rm(0))))

! Define height tolerance for Newton's method.

ztol = .05
lnmin = 10
if(( .not. fter ) .and. ( freq .le. 3001. )) lnmin = 9

! Initialize range and index variables for RO region.

xROn = 0.
iROp = -1

! Determine the minimum height the PE model must reach.

htest = htrap + hthick

if( ihybrid .eq. 1 ) then
   ztest = amax1( htest, 1.2*htermax )
else
   ztest = amax1( htlim, antref )
   hmx = ztest + antref + rmax**2 / aek2
   atest = atan( hmx / rmax )
   alflim = amax1( alflim, atest )
end if

! Now determine the maximum PE propagation angle needed to get
! to AT LEAST this height.  Also initialize all associated PE
! variables.

call getthmax( htest, htermax, rflat, ztest, alflim, THETAMAX )

if(( ihybrid .eq. 2 ) .and. ( zlim .gt. htlim )) ihybrid = 0

zlim = amin1( htlim, zlim )  !in case zlim > calculation height domain

! Maximize THETAMAX within determined FFT size for terrain cases and if
! using calculation modes IHYBRID=0 or 2.

if( fter ) then

! Use 74% of ZMAX instead of 75% to leave some slop and ensure the FFT size is
! not surpassed.

   if(( ihybrid .ne. 1 ) .and. ( .74*zmax .gt. zlim )) then
      thetafrac = alaunch / thetamax
      zmax = zlim / .74
      sthetamax = float(n) * wl * .5 / zmax

! Put upper limits on THETAMAX depending on frequency.
```

```
      if( freq .gt. 1000. ) then
         sthetamax = amin1( sthetamax, sdeg10 )
      else
         sthetamax = amin1( sthetamax, sdeg15 )
      end if
      delz= wl * .5 / sthetamax
      thetamax = asin( sthetamax )
      zmax = float(n) * delz
      alaunch = thetafrac * thetamax
      theta75 = .75 * thetamax
   end if
end if

! For calculation modes IHYBRID = 1 or 2, initialize all variables for use
! in XO calculations.

ixostp = 0
if( ihybrid .ne. 0 ) then

   if ( zlim .le. htlim-1.e-3 ) then

! Get bin # within calculation domain (with respect to HMINTER) at
! which to perform spectral estimation. From JZLIM to JZLIM-NPNTS.

      jzlim = int( zlim / delz )
      zlim = float( jzlim ) * delz

!Determine RATZ and IRATZ.

      j = iap
      id = 1
      do while( j .le. irtemp )
            if( htemp(j) .gt. zlim ) exit
            if( htemp(j) .gt. zrt(id) ) id = id + 1
            j = j + 1
      end do
      ira = amax0( 1, j-1 )
      idg = id - 1
      grd = gr(idg)

      rad = raya(ira)**2 + 2. * grd * ( zlim - htemp(ira) )
      aatz = 0.
      if( rad .gt. 0. ) aatz = sign( 1., raya(ira) ) * sqrt( rad )
      ratz = rtemp(ira) + (aatz - raya(ira)) / grd
      if(( ratz .lt. rmax ) .and. ( zlim .lt. htlim )) then
         k = 1
         do while( rngout(k) .lt. ratz )
            k = k + 1
         end do
         iratz = amin0( nrout, k )
         ixostp = iratz
      end if
   else
      iratz = nrout + 1
         ratz = 2. * rmax
   end if

end if
ixo = ixostp

! Determine horizon range based on transmitter height and 0 receiver height
! by RHOR = 3572. * sqrt( 1.3333 * antref )

rhor = 4124.5387 * sqrt( antht )
dr = fko2 * delz**2  !Just use this as a basis, DR may change later.
rkm = rmax * 1.e-3
```

```
! Determine PE range step and integer increment at which to store
! propagation factor, angle, and range at ZLIM.

if( fter ) then
   dr = amin1( dr, 700. )
   if( rkm .ge. 5. ) rllim = 75.
   if( rkm .ge. 10. ) rllim = 90.
   if( rkm .ge. 15. ) rllim = 100.
   if( rkm .ge. 20. ) rllim = 110.
   if( rkm .ge. 30. ) rllim = 175.
   if( rkm .ge. 50. ) rllim = 200.
   if( rkm .ge. 75. ) rllim = 250.
   if( rkm .ge. 100. ) rllim = 300.
   dr = amax1( dr, rllim )
   if( rfix .gt. 0. ) then
      rd = rfix / dr
      if( rd .lt. 1. ) then
         dr = nint( 1. / rd ) * rfix
      else
         dr = rfix / nint( rd )
      end if
   end if
   izinc = 1
else
   if( ihybrid .eq. 0 ) then
      dr = amin1( dr, 1000. )
      dr = amax1( dr, 30. )
      if( rmax .ge. rhor ) dr = amax1( 300., dr )
   end if
   izinc = 3
   if( freq .ge. 5000. ) izinc = 2
   if( freq .ge. 10000. ) izinc = 1
end if

! Determine number of points that will be stored in FFACZ(,), and
! allocate and initialize all arrays associated with of extended optics calcs.

if( ixostp .gt. 0 ) then
   rmxdif = rmax - ratz
   niz = nint( rmxdif / dr )
   izmax = niz / izinc + 4        !add some slop

! Initialize variables and filter array for spectral estimation.

   npnts = 8
   if( fter ) npnts= 16
   lnp = 6
   if( fter ) lnp = 7
   ns = 2**lnp
   np4 = npnts/4
   np34 = 3.* np4
   cnp75 = pi / np4

   call allarray_xo( IERROR )
   if( ierror .ne. 0 ) return

   do i = 0, np4
      fj= cnp75 * float(i)
      filtp(i) = .5 + .5 * cos(fj)
   end do
   xocon = wl / ns / 2. / delz
end if

dr2 = .5 * dr

! Initialize variables for free-space propagator phase calculations.
```

```
delp = pi/zmax
FNorm = 2. / N
cnst = delp / fko
nm1 = n - 1
dz2 = 2. * delz
n4 = n / 4

! Allocate and initialize all arrays associated with PE calcs.

call allarray_pe( IERROR )

! Initialize variables and set-up filter array for PE calculations.

n34 = 3.* n4
cn75 = pi / n4
do i = 0, n4
   fj= cn75 * float(i)
   filt(i) = .5 + .5 * cos(fj)
end do

! Initialize dielectric ground constants.

ig = 1
call dieinit
if(( freq .le. 300. ) .or. ( ipol .eq.1 )) call getaln

! Initialize starter field.

call xyinit

! Transform to z-space.

call fft( U )

! Initialize C1 and C2 for start of PE calculations

if( ipol .eq.1 ) then
   c1 = .5 * ( u(0) + u(n)*root(n) )
   c2 = .5 * ( u(0)*rootm(n) + u(n) )

   do i = 1, nm1
      c1c = u(i) * root(i)
      c2c = u(n-i) * rootm(i)

      c1 = c1 + c1c
      c2 = c2 + c2c
   end do
   c1 = c1 * rk
   c2 = c2 * rk

end if

ylast = 0.
if( fter ) ylast = ty(1)

ycurm = 0.
ycur = 0.

! Define mesh array in height

do i=0,n
   ht(i)= float(i)*delz
end do

iz = 1
```

```
! Now fill height array separating flat earth region from RO region.
! The height is determined and stored at each output range step.

call fillht

! Determine the free-space propagator (p-space) arrays.

call phase1

! For special case when ground is initially flat, but at non-zero
! height, re-adjust all refractivity arrays.

if(( ihybrid .eq. 1 ) .and. ( abs(ty(1)) .gt. 1.e-3 ) .and. (fter)) then
   nlevel = levels
   yref = ty(1)
   href = 0.
   refref = 0.
   js = -1

! Get refractivity profile level at which the height of the ground is just
! above. This level is JS.

   do i = 0, nlevel
      if(( yref .le. zrt(i+1) ) .and. ( yref .gt. zrt(i) )) js = i
   end do

! Determine the refractivity value at the ground and fill arrays HREF() and
! REFREF() with refractivity profile where height 0. now refers to the ground
! reference,i.e., either local ground height or HMINTER.

   if( js .gt. -1 ) then
      jsp1 = js + 1
      frac = (yref - zrt(js))/(zrt(jsp1) - zrt(js))
      rmu = rm(js) + frac * (rm(jsp1) - rm(js))
      if( int( frac ) .eq. 1 ) js = jsp1
      newl = nlevel - js
      refref(0) = rmu
      href(0) = 0.
      k = js + 1
      do jk = 1, newl
         refref(jk) = rm(k)
         href(jk) = zrt(k) - yref
         k = k + 1
      end do
      levels = newl
      do i = 0, levels
         rm(i) = refref(i)
         zrt(i) = href(i)
      end do
   end if

   do i = 0, levels
      ip1 = i + 1
      rmd = rm(ip1) - rm(i)
      g = rmd / (zrt(ip1) - zrt(i))
      if( abs( g ) .lt. 1.e-8 ) g = sign( 1., g )*1.e-8
      gr(i) = g
      q(i) = 2. * rmd
   end do
end if

! If smooth surface and range-independent case then initialize all refractivity
! and z-space propagator arrays now.

if((.not. fter) .and. ( nprof .eq. 1 )) then
   call intprof
   call phase2
```

```
end if

if( kabs .eq. 1 ) call gasabs
if( kabs .eq. 2 ) gasatt = gammaa * 1.e-2

end subroutine apminit
```

A.1.1 Subroutine ALLARRAY_APM

```
!********************** SUBROUTINE ALLARRAY_APM ***********************

! Module Name: ALLARRAY_APM

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine allocates and initializes all dynamically dimensioned
!          arrays associated with APM general info, terrain, refractivity,
!                and troposcatter arrays.

! Version Number: 1.0

! INPUTS:
!   Argument List: None
!   Common: IGR, ITPA, ITROPO, LVLP, NFACS, NROUT, NZOUT

! OUTPUTS:
!   Argument List: IERROR
!   Common: None
!   Public: AD1(), ADIF(), CN2(), D2S(), DIELEC(,), FSLR(), GR(),
!       HFANGR(), HLIM(), HREF(), HTDUM(), HTFE(), IGRND(), Q(), RDT(),
!       REFDUM(), REFREF(), RFAC1(), RFAC2(), RGRND(), RLOGO(), RLOSS(), RM(),
!       RNGOUT(), RSQRD(), SLP(), TH1(), THETA0(), THETA2S(), TX(),
!        TY(), ZOUT(), ZOUTMA(), ZOUTPA(), ZRO(), ZRT()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines called:
!   APM Specific: NONE
!   Intrinsic: ALLOCATE, ALLOCATED, DEALLOCATE

! GLOSSARY: See universal glossary for common and public variables.

!   Input Variables: None

!   Output Variables:
!      IERROR = Integer variable indicating error # for DEALLOCATE and
!               ALLOCATE statements.

!   Local Variables:
!              LVLPT = LVLP + 1 -> upper boundary limit on array G

subroutine allarray_apm( IERROR )

use apm_mod

ierror = 0

if( nfacs .gt. 0 ) then
   IF( ALLOCATED( HFANGR ) ) DEALLOCATE( HFANGR, stat=ierror )
   ALLOCATE( HFANGR(NFACS), stat=ierror )
   if( ierror .ne. 0 ) return
   HFANGR = 0.
end if
```

```
if( allocated( rsqrd ) ) deallocate( rsqrd, stat=ierror )
allocate( rsqrd(nrout), stat=ierror )
if( ierror .ne. 0 ) return
rsqrd = 0.

if( allocated( fslr ) ) deallocate( fslr, stat=ierror )
allocate( fslr(nrout), stat=ierror )
if( ierror .ne. 0 ) return
fslr = 0.

if( allocated( rlogo ) ) deallocate( rlogo, stat=ierror )
allocate( rlogo(nrout), stat=ierror )
if( ierror .ne. 0 ) return
rlogo = 0.

if( allocated( rngout ) ) deallocate( rngout, stat=ierror )
allocate( rngout(nrout), stat=ierror )
if( ierror .ne. 0 ) return
rngout = 0.

if( allocated( zout ) ) deallocate( zout, stat=ierror )
allocate( zout(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
zout = 0.

if( allocated( zro ) ) deallocate( zro, stat=ierror )
allocate( zro(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
zro = 0.

if( allocated( zoutma ) ) deallocate( zoutma, stat=ierror )
allocate( zoutma(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
zoutma = 0.

if( allocated( zoutpa ) ) deallocate( zoutpa, stat=ierror )
allocate( zoutpa(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
zoutpa = 0.

if( allocated( hlim ) ) deallocate( hlim, stat=ierror )
allocate( hlim(nrout), stat=ierror )
if( ierror .ne. 0 ) return
hlim = 0.

if( allocated( htfe ) ) deallocate( htfe, stat=ierror )
allocate( htfe(nrout), stat=ierror )
if( ierror .ne. 0 ) return
htfe = 0.

if( allocated( rfac1 ) ) deallocate( rfac1, stat=ierror )
allocate( rfac1(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
rfac1 = 0.

if( allocated( rfac2 ) ) deallocate( rfac2, stat=ierror )
allocate( rfac2(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
rfac2 = 0.

if( allocated( rloss ) ) deallocate( rloss, stat=ierror )
allocate( rloss(0:nzout), stat=ierror )
if( ierror .ne. 0 ) return
rloss = 0.

! Allocate arrays associated with terrain info.
```

```
if( allocated( tx ) ) deallocate( tx, stat=ierror )
allocate( tx(itpa), stat=ierror )
if( ierror .ne. 0 ) return
tx = 0.

if( allocated( ty ) ) deallocate( ty, stat=ierror )
allocate( ty(itpa), stat=ierror )
if( ierror .ne. 0 ) return
ty = 0.

if( allocated( slp ) ) deallocate( slp, stat=ierror )
allocate( slp(itpa), stat=ierror )
if( ierror .ne. 0 ) return
slp = 0.

if( igr .eq. 0 ) then
   igr = 1
   IF( ALLOCATED( DIELEC ) ) DEALLOCATE( DIELEC, stat=ierror )
   ALLOCATE( DIELEC(2, IGR), stat=ierror )
   if( ierror .ne. 0 ) return
   DIELEC = 0.

   IF( ALLOCATED( IGRND ) ) DEALLOCATE( IGRND, stat=ierror )
   ALLOCATE( IGRND(IGR), stat=ierror )
   if( ierror .ne. 0 ) return
   IGRND = 0

   IF( ALLOCATED( RGRND ) ) DEALLOCATE( RGRND, stat=ierror )
   ALLOCATE( RGRND(IGR), stat=ierror )
   if( ierror .ne. 0 ) return
   RGRND = 0.
end if

IF( ALLOCATED( cn2 ) ) DEALLOCATE( cn2, stat=ierror )
ALLOCATE( cn2(IGR), stat=ierror )
if( ierror .ne. 0 ) return
cn2 = cmplx(0., 0.)

! Allocate arrays associated with refractivity info.

if( allocated( refdum ) ) deallocate( refdum, stat=ierror )
allocate( refdum(0:lvlp), stat=ierror )
if( ierror .ne. 0 ) return
refdum = 0.

if( allocated( htdum ) ) deallocate( htdum, stat=ierror )
allocate( htdum(0:lvlp), stat=ierror )
if( ierror .ne. 0 ) return
htdum = 0.

if( allocated( href ) ) deallocate( href, stat=ierror )
allocate( href(0:lvlp), stat=ierror )
if( ierror .ne. 0 ) return
href = 0.

if( allocated( refref ) ) deallocate( refref, stat=ierror )
allocate( refref(0:lvlp), stat=ierror )
if( ierror .ne. 0 ) return
refref = 0.

lvlpt = lvlp + 1
if( allocated( gr ) ) deallocate( gr, stat=ierror )
allocate( gr(0:lvlpt), stat=ierror )
if( ierror .ne. 0 ) return
gr = 0.
```

```
if( allocated( q ) ) deallocate( q, stat=ierror )
allocate( q(0:lvlpt), stat=ierror )
if( ierror .ne. 0 ) return
q = 0.

if( allocated( rm ) ) deallocate( rm, stat=ierror )
allocate( rm(0:lvlpt), stat=ierror )
if( ierror .ne. 0 ) return
rm = 0.

if( allocated( zrt ) ) deallocate( zrt, stat=ierror )
allocate( zrt(0:lvlpt), stat=ierror )
if( ierror .ne. 0 ) return
zrt = 0.

!Initialize and allocate arrays associated with troposcatter calculations.

if( itropo .eq. 1 ) then

   if( allocated( adl ) ) deallocate( adl, stat=ierror )
   allocate( adl(itpa), stat=ierror )
   if( ierror .ne. 0 ) return
   adl = 0.

   if( allocated( adif ) ) deallocate( adif, stat=ierror )
   allocate( adif(0:nzout), stat=ierror )
   if( ierror .ne. 0 ) return
   adif = 0.

   if( allocated( d2s ) ) deallocate( d2s, stat=ierror )
   allocate( d2s(0:nzout), stat=ierror )
   if( ierror .ne. 0 ) return
   d2s = 0.

   if( allocated( rdt ) ) deallocate( rdt, stat=ierror )
   allocate( rdt(0:nzout), stat=ierror )
   if( ierror .ne. 0 ) return
   rdt = 0.

   if( allocated( thl ) ) deallocate( thl, stat=ierror )
   allocate( thl(itpa), stat=ierror )
   if( ierror .ne. 0 ) return
   thl = 0.

   if( allocated( theta0 ) ) deallocate( theta0, stat=ierror )
   allocate( theta0(nrout), stat=ierror )
   if( ierror .ne. 0 ) return
   theta0 = 0.

   if( allocated( theta2s ) ) deallocate( theta2s, stat=ierror )
   allocate( theta2s(0:nzout), stat=ierror )
   if( ierror .ne. 0 ) return
   theta2s = 0.

end if

end subroutine allarray_apm
```

A.1.2 Subroutine ALLARRAY_PE

```
!********************** SUBROUTINE ALLARRAY_PE ***********************

! Module Name: ALLARRAY_PE

! Module Security Classification: UNCLASSIFIED
```

```
! Purpose: This routine allocates and initializes all dynamically
!          dimensioned arrays associated with PE calculations.

! Version Number: 1.0

! INPUTS:
!   Argument List: None
!   Common: N, N4

! OUTPUTS:
!   Argument List: IERROR
!   Common: None
!   Public: ENVPR(), FILT(), FRSP(), HT(), PROFINT(), ROOT(), ROOTM(),
!           U(), ULST(), W(), XDUM(), YDUM(), YM()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines called:
!   APM Specific: NONE
!   Intrinsic: ALLOCATE, ALLOCATED, DEALLOCATE

! GLOSSARY: See universal glossary for common and public variables.

!   Input Variables: None

!   Output Variables:
!      IERROR = Integer variable indicating error # for DEALLOCATE and
!               ALLOCATE statements.

!   Local Variables: None

subroutine allarray_pe( IERROR )

use apm_mod

ierror = 0

if( allocated( root ) ) deallocate( root, stat=ierror )
allocate( root(0:n), stat=ierror )
if( ierror .ne. 0 ) return
root = cmplx( 0., 0.)

if( allocated( rootm ) ) deallocate( rootm, stat=ierror )
allocate( rootm(0:n), stat=ierror )
if( ierror .ne. 0 ) return
rootm = cmplx( 0., 0.)

if( allocated( envpr ) ) deallocate( envpr, stat=ierror )
allocate( envpr(0:n), stat=ierror )
if( ierror .ne. 0 ) return
envpr = cmplx( 0., 0.)

if( allocated( frsp ) ) deallocate( frsp, stat=ierror )
allocate( frsp(0:n), stat=ierror )
if( ierror .ne. 0 ) return
frsp = cmplx( 0., 0.)

if( allocated( u ) ) deallocate( u, stat=ierror )
allocate( u(0:n), stat=ierror )
if( ierror .ne. 0 ) return
u = cmplx( 0., 0.)

if( allocated( ulst ) ) deallocate( ulst, stat=ierror )
allocate( ulst(0:n), stat=ierror )
```

```
if( ierror .ne. 0 ) return
ulst = cmplx( 0., 0.)

if( allocated( filt ) ) deallocate( filt, stat=ierror )
allocate( filt(0:n4), stat=ierror )
if( ierror .ne. 0 ) return
filt = 0.

if( allocated( ht ) ) deallocate( ht, stat=ierror )
allocate( ht(0:n), stat=ierror )
if( ierror .ne. 0 ) return
ht = 0.

if( allocated( profint ) ) deallocate( profint, stat=ierror )
allocate( profint(0:n), stat=ierror )
if( ierror .ne. 0 ) return
profint = 0.

if( allocated( xdum ) ) deallocate( xdum, stat=ierror )
allocate( xdum(0:n), stat=ierror )
if( ierror .ne. 0 ) return
xdum = 0.

if( allocated( ydum ) ) deallocate( ydum, stat=ierror )
allocate( ydum(0:n), stat=ierror )
if( ierror .ne. 0 ) return
ydum = 0.

if( allocated( w ) ) deallocate( w, stat=ierror )
allocate( w(0:n), stat=ierror )
if( ierror .ne. 0 ) return
w = cmplx( 0., 0. )

if( allocated( ym ) ) deallocate( ym, stat=ierror )
allocate( ym(0:n), stat=ierror )
if( ierror .ne. 0 ) return
ym = cmplx( 0., 0. )

end subroutine allarray_pe
```

A.1.3 Subroutine ALLARRAY_XO

```
!********************** SUBROUTINE ALLARRAY_XO ***********************

! Module Name: ALLARRAY_XO

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine allocates and initializes all dynamically
!          dimensioned arrays associated with XO calculations.

! Version Number: 1.0

! INPUTS:
!   Argument List: None
!   Common: IZMAX, NP4, NROUT, NS

! OUTPUTS:
!   Argument List: None
!   Common: None
!   Public: FFROUT(,), FILTP(), GRAD(), HTR(), LVL(), SPECTR(), XP(), YP()

! Modules Used: APM_MOD

! Calling Routines: APMINIT
```

```
! Routines called:
!   APM Specific: NONE
!   Intrinsic: ALLOCATE, ALLOCATED, DEALLOCATE

! GLOSSARY: See universal glossary for common and public variables.

!   Input Variables: None

!   Output Variables:
!      IERROR = Integer variable indicating error # for DEALLOCATE and
!               ALLOCATE statements.

!   Local Variables: None

subroutine allarray_xo( IERROR )

use apm_mod

ierror = 0

if( allocated( ffrout ) ) deallocate( ffrout, stat=ierror )
allocate( ffrout(2,nrout), stat=ierror )
if( ierror .ne. 0 ) return
ffrout = 0.

if( allocated( ffacz ) ) deallocate( ffacz, stat=ierror )
allocate( ffacz(3,izmax), stat=ierror )
if( ierror .ne. 0 ) return
ffacz = 0.

if( allocated( grad ) ) deallocate( grad, stat=ierror )
allocate( grad(0:lvlp,izmax), stat=ierror )
if( ierror .ne. 0 ) return
grad = 0.

if( allocated( htr ) ) deallocate( htr, stat=ierror )
allocate( htr(0:lvlp,izmax), stat=ierror )
if( ierror .ne. 0 ) return
htr = 0.

if( allocated( lvl ) ) deallocate( lvl, stat=ierror )
allocate( lvl(izmax), stat=ierror )
if( ierror .ne. 0 ) return
lvl = 0.

! Allocate and initialize all arrays associated with spectral estimation
! of PE field.

if( allocated( filtp ) ) deallocate( filtp, stat=ierror )
allocate( filtp(0:np4), stat=ierror )
if( ierror .ne. 0 ) return
filtp = 0.

if( allocated( xp ) ) deallocate( xp, stat=ierror )
allocate( xp(0:ns), stat=ierror )
if( ierror .ne. 0 ) return
xp = 0.

if( allocated( yp ) ) deallocate( yp, stat=ierror )
allocate( yp(0:ns), stat=ierror )
if( ierror .ne. 0 ) return
yp = 0.

if( allocated( spectr ) ) deallocate( spectr, stat=ierror )
allocate( spectr(0:ns), stat=ierror )
if( ierror .ne. 0 ) return
```

```
spectr = 0.

end subroutine allarray_xo
```

A.1.4 Subroutine ANTPAT

```
! ************************** SUBROUTINE ANTPAT ******************************

! Module Name: ANTPAT

! Module Security Classification: UNCLASSIFIED

! Purpose: Determines the antenna pattern factor for angle passed to routine.

! Version Number: 1.0

! INPUTS:
!   Argument List: ANG
!   Common: AFAC, ALPHAD, BW, ELV, IPAT, NFACS, PELEV, SBW, UMAX
!   Public: HFANGR(), HFFAC()

! OUTPUTS:
!   Argument List: PATFAC
!   Common: NONE

! Modules Used: APM_MOD

! Calling Routines: FEM, ROCALC, TROPO, TROPOINIT, XYINIT

! Routines called:
!   APM Specific: NONE
!   Intrinsic: ABS, AMAX1, AMIN1, EXP, SIN

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ANG = elevation angle at transmitter

!   Output Variables:
!       PATFAC = antenna pattern factor for angle ANG

!   Local Variables:
!       UDIF = Angle relative to the elevation angle of the main beam.

subroutine antpat( ang, PATFAC )

use apm_mod

! IPAT = 1 gives Omnidirectional antenna pattern factor : f(u) = 1
! Default for Omni antenna pattern

patfac = 1.

! In the following pattern definitions, "ua" refers to the angle for which
! the antenna pattern is sought, and "u0" refers to the elevation angle.

select case( ipat )

   case( 2 )

!  IPAT = 2 gives Gaussian antenna pattern based on
!  f(p-p0) = exp(-w**2 * ( p-p0 )**2 ) / 4, where p = sin(u) and
!  p0 = sin(u0)

      pr = sin(ang) - pelev
```

```
      patfac = exp(-pr * pr * afac)

   case( 4 )

! IPAT = 4 gives csc-sq pattern based on
! f(u) = 1 for ua-u0 <= bw
! f(u) = sin(bw) / sin(ua-u0) for ua-u0 > bw
! f(u) = maximum of .03 or [1+(ua-u0)/bw] for ua-u0 < 0

      udif = ang - elv
      if( udif .gt. bw ) then
         patfac = sbw / sin( udif )
      elseif( udif .lt. 0 ) then
         patfac = amin1( 1., amax1( .03, (1. + udif/bw) ) )
      end if

   case( 3, 5, 6 )

! IPAT = 3 gives sin(x)/x pattern based on
! f(ua-u0) = sin(x) / x where x = afac * sin(ua-u0) for |ua-u0| <= umax
! f(ua-u0) = 0 for |ua-u0| > umax
! IPAT = 5 gives height-finder pattern which is a special case of sin(x)/x

      udif = ang - elv
      if( ipat .ge. 5 ) then
         chi = elv
         if( alphad .gt. elv ) then
            udif = ang - alphad
            chi = alphad
         end if
      end if

      if( abs(udif) .le. 1.e-6 ) then
         patfac = 1.
      elseif( abs( udif ) .gt. umax ) then
         patfac = 0.
      else
         arg = afac * sin( udif )
         patfac = sin( arg ) / arg
      end if

! For IPAT = 6, user-specified height-finder antenna pattern,
! adjust user-defined height-finder pattern by appropriate factor
! based on HFANGR() and HFFAC() arrays.  Adjustment is not necessary
! when CHI is less than the first user-defined angle (HFANGR(1)).

      IF( ipat .EQ. 6 ) then
            if( chi .GT. hfangr(1) ) THEN
            i = nfacs
            DO WHILE (chi .LE. hfangr(i))
               i = i - 1
            END DO
            patfac = patfac * hffac(i)
         end if
      END IF

   case default

      ! do nothing

end select

end subroutine antpat
```

A.1.5 Subroutine DIEINIT

```
! ************************* SUBROUTINE DIEINIT *****************************

! Module Name: DIEINIT

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine calculates conductivity and permittivity
!          as a function of frequency in MHz.  All equations and coef-
!          ficients were obtained by using a SUMMASKETCH digitizer to digitize
!          the CCIR volume 5 curves on page 74.  The digitized data were
!          then used with TABLECURVE software to obtain the best fit
!          equations and coefficients used in this subroutine.  In some
!          cases two sets of equations were required to obtain a decent
!          fit across the 100 MHz - 100GHz range.  These curves fit the
!          digitized data to within 5%.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: FREQ, IGR
!   Public: DIELEC(,), IGRND(), RGRND()

! OUTPUTS:
!   Argument List: NONE
!   Public: CN2()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines called:
!   APM Specified: NONE
!   Intrinsic: DBLE, SQRT

! GLOSSARY:

!   Input Variables:
!       See universal glossary for common and public variables

!   Output Variables:
!       See universal glossary for common and publicvariables

!   Local Variables:
!       EPSILON = relative permittivity
!       SIGMA = conductivity
!       F1-8 = Frequency in MHz to the nth power. I.e., f5 = freq**5
!       A() thru F() = polynomial coefficients for use in determining
!           EPSILON and SIGMA.

subroutine dieinit

use apm_mod

dimension a(18), b(18), c(18), d(18), e(18), f(18)

double precision f1, f2, f3, f4, f5, f6, f7, f8, f9

data (a(i),i=1,18) / 1.4114535e-2, 3.8586749, 79.027635,    &
                    -0.65750351, 201.97103, 857.94335,           &
                     915.31026, 0.8756665, 5.5990969e-3,    &
                     215.87521, .17381269, 2.4625032e-2,    &
                    -4.9560275e-2, 2.2953743e-4, .000038814567, &
                     1.2434792E-04, 51852.543, 4.13105E-05 /
```

```
data (b(i),i=1,18) / -5.2122497e-8, -2.1179295e-5, -2.2083308e-5,  &
                     5.5620223e-5, -2.5539582e-3, -8.9983662e-5,   &
                     -9.4530022e-6, 4.7236085e-5, 8.7798277e-5,    &
                     -7.6649237e-5, 1.2655183e-4, 1.8254018e-4,     &
                     2.9876572e-5, -8.1212741e-7, 8.467523E-02,    &
                     2.824598E-04, 3.883854E-02, 2.03589E-07 /
data (c(i),i=1,18) / 5.8547829e-11, 9.1253873e-4, -3.5486605e-4,   &
                     6.6113198e-4, 1.2197967e-2, 5.5275278e-2,        &
                     -4.0348211e-3, 2.6051966e-8, 6.2451017e-8,       &
                     -2.6151055e-3, -1.6790756e-9, -2.664754e-8,   &
                     -3.0561848e-10, 1.8045461e-9, 9.878241E-06,   &
                     8.680839E-07, 389.58894, -3.1739E-12 /
data (d(i),i=1,18) / -7.6717423e-16, 6.5727504e-10, 2.7067836e-9,  &
                     3.0140816e-10, 3.7853169e-5, 8.8247139e-8,    &
                     4.892281e-8, -9.235936e-13, -7.1317207e-12,   &
                     1.2565999e-8, 1.1037608e-14, 7.6508732e-12,   &
                     1.1131828e-15, -1.960677e-12, -9.736703E-05,  &
                     -6.755389E-08, 6.832108E-05, 4.52331E-17 /
data (e(i),i=1,18) / 2.9856318e-21, 1.5309921e-8, 8.210184e-9,        &
                     1.4876952e-9, -1.728776e-6, 0.0,                         &
                     7.4342897e-7, 1.4560078e-17, 4.2515914e-16,   &
                     1.9484482e-7, -2.9223433e-20, -7.4193268e-16, &
                     0.0, 1.2569594e-15, 7.990284E-08,             &
                     7.2701689e-11, 0., 0.  /
data (f(i),i=1,18) / 0., -1.9647664e-15, -1.0007669e-14, 0., 0.,   &
                     0., 0., -1.1129348e-22, -1.240806e-20, 0.,    &
                     0., 0., 0., -4.46811e-19, 3.269059E-07,       &
                     2.8728975e-12, 0., 0. /

f1 = dble( freq )
f2 = f1 * f1
f3 = f1 * f2
f4 = f1 * f3
f5 = f1 * f4
f6 = f1 * f5
f7 = f1 * f6
f8 = f1 * f7
f9 = f1 * f8

do i = 1, igr

   select case ( igrnd(i) )

      case( 0 )  ! Permittivity and conductivity for salt water
         epsilon = 70.
         sigma = 5.
         m = 1
         m1 = m + 1
         if( f1 .gt. 2253.5895 ) epsilon = 1. / ( a(m) + b(m)*f1   &
            + c(m)*f2 + d(m)*f3 + e(m)*f4 )
         if( f1 .gt. 1106.207 ) then
            sigma = a(m1) + c(m1)*f1 + e(m1)*f2
            sigma = sigma / ( 1.+ b(m1)*f1 + d(m1)*f2 + f(m1)*f3 )
         end if

      case( 1 )  !Permittivity and conductivity for fresh water
         epsilon = 80.0
         m = 3
         m1 = m + 1
         IF( f1 .gt. 6165.776 ) THEN
            epsilon = a(m) + c(m)*f1 + e(m)*f2
            epsilon = epsilon/(1. + b(m)*f1 + d(m)*f2 + f(m)*f3 )
         end if
         IF( f1 .gt. 5776.157) THEN
            k = 2
         else
            m1 = m1 + 1
```

```
      k = -1
   end if
   sigma = a(m1) + c(m1)*f1 + e(m1)*f2
   sigma = (sigma / (1. + b(m1)*f1 + d(m1)*f2))**k

case( 2 ) !Permittivity and conductivity for wet ground
   epsilon = 30.0
   m = 6
   IF( f1 .ge. 4228.11 ) m = 7
   if( f1 .gt. 1312.054 ) then
      epsilon = a(m) + c(m)*f1 + e(m)*f2
      epsilon = SQRT( epsilon / (1. + b(m)*f1 + d(m)*f2) )
   end if
   IF( f1 .gt. 15454.4) then
      m1 = 8
      g = 3.3253339e-28
   else
      m1 = 9
      g = 1.3854354e-25
   end if
   sigma = a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3 + e(m1)*f4
   sigma = sigma + f(m1)*f5 + g*f6

case( 3 ) !Permittivity and conductivity for medium dry ground
   epsilon = 15.0
   IF( f1 .gt. 4841.945) THEN
      m = 10
      epsilon = a(m) + c(m)*f1 + e(m)*f2
      epsilon = SQRT( epsilon / (1. + b(m)*f1 + d(m)*f2) )
   end if
   m1 = 12
   IF( f1 .gt. 4946.751) m1 = 11
   sigma = (a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3 + e(m1)*f4)**2

case( 4 ) !Permittivity and conductivity for very dry ground
   epsilon = 3.0
   IF( f1 .lt. 590.8924 ) then
      sigma = 1.0e-4
   else
      IF( f1 .gt. 7131.933) THEN
         m1 = 13
         sigma = (a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3)**2
      else
         m1 = 14
         g = 9.4623158e-23
         h = -1.1787443e-26
         s = 7.9254217e-31
         t = -2.2088286e-35
         sigma = a(m1) + b(m1)*f1 + c(m1)*f2 + d(m1)*f3
         sigma = sigma + e(m1)*f4 + f(m1)*f5 + g*f6
         sigma = sigma + h*f7 + s*f8 + t*f9
      end if
   end if

case( 5 ) !Permittivity and conductivity for ice at -1 degree C
   epsilon = 3.0
   IF (f1 .le. 300.0) THEN
      m = 15
         signum = a(m) + c(m) * f1 + e(m) * f2
         sigdnom = 1.0 + b(m) * f1 + d(m) * f2 + f(m) * f3
       ELSE
         m = 16
         g = -2.6416983e-14
         h = -1.8795958e-18
         si = 1.37552E-18
         signum = a(m) + c(m)*f1 + e(m)*f2 + g*f3 + si*f4
      sigdnom = 1.0 + b(m)*f1 + d(m)*f2 + f(m)*f3 + h*f4
```

```
            END IF
         sigma = signum / sigdnom

      case( 6 ) !Permittivity and conductivity for ice at -10 degrees C
         epsilon = 3.0
         IF( f1 .le. 8753.398) THEN
            m = 17
               sigma = 1.0 / ((a(m) + c(m)*f1) / (1.0 + b(m)*f1 + d(m)*f2))
            ELSE
            m = 18
               sigma = a(m) + b(m)*f1 + c(m)*f2 + d(m)*f3
            END IF

       case( 7 )
         epsilon = dielec(1,i)
         sigma = dielec(2,i)

      case default
         ! Do nothing
   end select

   s1 = sigma * 60. * wl
   cn2(i) = cmplx( epsilon, s1 )

end do

end subroutine dieinit
```

A.1.6 Subroutine FFT

```
! *************************** SUBROUTINE FFT ********************************

! Module Name: FFT

! Module Security Classification: UNCLASSIFIED

! Purpose: Performs fast Fourier sine transform on complex array U.

! Version Number: 1.0

! INPUTS:
!   Argument List: UXY()
!   Common: LN, N

! OUTPUTS:
!   Argument List: UXY()
!   Common: NONE
!   Public: XDUM(), YDUM()

! Modules Used: APM_MOD

! Calling Routines: APMINIT, FRSTP

! Routines called:
!   APM Specific: SINFFT
!   Intrinsic: REAL, IMAG, CMPLX

! GLOSSARY: See universal glossary for common variables and parameters.

!   Input Variables:
!       UXY() = Complex field to be transformed.

!   Output Variables:
!       UXY() = Transform of complex field.
```

```
subroutine fft( UXY )

use apm_mod

complex uxy(0:*)

do i = 0, n
   xdum(i) = real( uxy(i) )
   ydum(i) = imag( uxy(i) )
end do

call sinfft( ln, XDUM )
call sinfft( ln, YDUM )

do i = 0, n
   uxy(i) = cmplx( xdum(i), ydum(i) )
end do

end subroutine fft
```

A.1.7 Subroutine FFTPAR

```
! *********************** SUBROUTINE FFTPAR **********************

! Module Name: FFTPAR

! Module Security Classification: UNCLASSIFIED

! Purpose: Determines and computes the FFT size needed for a given
!          problem, plus all other associated PE variables.

! Verion Number: 1.0

! INPUTS:
!   Argument List: IFLAG, LNMIN, THETAMAX, WL
!   Common: NONE

! OUTPUTS:
!   Argument List: DELZ, LN, N, ZLIM, ZMAX
!   Common: NONE

! Calling Routines: GETTHMAX

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: FLOAT, SIN

! GLOSSARY:

!   Input Variables:
!       IFLAG = flag indicating whether to determine maximum FFT size
!               based on given THETAMAX and height needed to reach (ZLIM),
!               or determine maximum height ZLIM based on given THETAMAX
!               and FFT size.
!               = 0 -> determine N, LN given THETAMAX and ZLIM
!               = 1 -> determine ZLIM given THETAMAX and LN
!       LNMIN = Minimum power of 2 transform size.  LNMIN = 9 for smooth
!               surface and frequencies <= 3000 MHz.  LNMIN = 10 all other
!               cases.
!       THETAMAX = Maximum PE propagation angle in radians.
!       WL = Wavelength in meters

!   Output Variables:
!       DELZ = Bin width in z-space = WL / (2*sin(THETAMAX))
!       LN = Power of 2 transform size, i.e. N = 2**LN
```

```
!        N = Transform size
!        ZLIM = Maximum internal height (HTLIM) or .75*ZMAX, whichever
!               is smaller.
!        ZMAX = Maximum height of PE calculation domain = N * DELZ

!    Local Variables:
!        STHETAMAX = Sine of THETAMAX.
!        ZT = ZLIM with a "fudge" factor.

subroutine fftpar( lnmin, wl, thetamax, iflag, ZLIM, ZMAX, DELZ, LN, N )

sthetamax = sin( thetamax )
delz= wl * .5 / sthetamax

if( iflag .eq. 0 ) then

! Set lower FFT limit to 2**10 if terrain profile or IHYBRID = 1, otherwise,
! lower limit is 2**9.

   ln = lnmin
   N=2**LN
   zmax=delz*float(n)

! Determine transform size needed to perform calculations to a height of ZTEST.

   zt = zlim - 1.e-3
   do while( .75*zmax .lt. zt )
      ln = ln + 1
      n = 2**ln
      zmax = delz * float(n)
   end do

elseif( iflag .eq. 1 ) then

! Determine the maximum height that can be reached given THETAMAX
! and LN.

   N=2**LN
   zmax=delz*float(n)

end if

zlim = .75 * zmax

end subroutine fftpar
```

A.1.8 Subroutine FILLHT

```
!************************* SUBROUTINE FILLHT *************************

! Module Name: FILLHT

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine calculates the effective earth radius for an
!          initial launch angle of 5 degrees.  Then it fills the array
!          HTFE() with height values of the limiting sub-model (depending
!          on value of IHYBRID) at each output range.  I.e., if
!          IHYBRID = 1, then HTFE() will contain height values at each
!          output range separating the FE region from the RO region.
!          If IHYBRID = 0 or 2, then HTFE() will contain those height
!          values at each output range at which the initial launch angle
!          has been traced to the ground or surface. These height values
!          represent the separating region where, above that height, valid
!          loss is computed, and below that height, no loss is computed
```

```
!          (outside PE angle limit).

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: ALAUNCH, ANTHT, ANTREF, HMREF, HTLIM, IHYBRID, ISTART, LEVELS
!           NROUT, YFREF
!   Public: GR(), RNGOUT(), ZRT()
!   Data: RTST

! OUTPUTS:
!   Argument List: NONE
!   Common: TWOKA
!   Public: HTFE()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: AMAX1, AMIN1

! GLOSSARY:

!   Input Variables: See universal glossary for common variables.

!   Output Variables: See universal glossary for common variables.

!   Local Variables:
!       A0 = Angle at start of trace in radians.
!       A1 = Angle at end of trace in radians.
!       GRD = Gradient of current refractivity layer being traced through.
!       H0 = Height at start of trace in meters.
!       H1 = Height at end of trace in meters.
!       H5 = Height of 5 degree ray traced to each output range point
!       IQUIT = Flag to end loop.
!       JL = Index in indicating location of source height in array ZRT()
!       R0 = Range at start of trace in meters.
!       R1 = Range at end of trace in meters.
!       RO = Current output range to trace to.
!       YAR = Height of image source.

subroutine fillht

use apm_mod

data a5 / 0.087266 /  !5 degrees in radians
data tan5 / 8.748866353e-2 /   ! tangent of 5 degrees

! Define in line ray trace functions:

radal( a, b ) = a**2 + 2. * grd * b              !a=a0, b=h1-h0
rp( a, b ) = a + b / grd                         !a=r0, b=a1-a0
ap( a, b ) = a + b * grd                         !a=a0, b=r1-r0
hp( a, b, c ) = a + ( b**2 - c**2 ) / 2. / grd   !a=h0, b=a1, c=a0

if( ihybrid .eq. 1 ) then

!Trace 5 degree elevation angle ray up to maximum height HTLIM to define
!effective earth radius. Then compute twoka (2*ek*a) for use in FEM to
!correct heights for earth curvature and average refraction.

   a0 = a5
   r0 = 0.
   i = 0
```

```
      DO WHILE ((zrt(i + 1) .LT. htlim) .AND. (i .LT. levels))
         grd = gr(i)
         rad = radal( a0, zrt(i+1)-zrt(i) )
         a1 = sqrt( rad )
         r1 = rp( r0, a1-a0 )
         a0 = a1
         r0 = r1
         i = i + 1
      END DO
      grd = gr(i)
      rad = radal( a0, htlim-zrt(i) )
      a1 = sqrt( rad )
      r1 = rp( r0, a1-a0 )

      twoka = (r1 ** 2) / (htlim - a5 * r1)

! Fill height array separating FE region from RO region.

      yar = yfref - antht
      do i = 1, nrout
         htfe(i) = yfref
         if( rngout(i) .gt. rtst ) then
            h5 = amax1( yfref, yar + tan5 * rngout(i) )
            htfe(i) = amin1( htlim, h5 )
         end if
      end do

   else

! For PE+XO or PE running modes, trace initial launch angle until it
! hits ground, storing heights traced at each output range.

      a0 = -alaunch
      r0 = 0.
      h0 = antref
      jl = istart
      iquit = 0
      jr = 1

      do while(( iquit .eq. 0 ) .or. ( jr .le. nrout ))
         htfe(jr) = 0.
         ro = rngout(jr)

         do while(( r0 .lt. ro ) .and. ( iquit .eq. 0 ))

            r1 = ro
            if( a0 .lt. 0 ) grd = gr(jl - 1)
            a1 = ap( a0, r1-r0 )
            h1 = hp( h0, a1, a0 )

            if( a1 .lt. 0. ) then
               if( h1 .lt. zrt(jl-1)+1.e-3 ) then
                  jl = jl - 1
                  h1 = zrt(jl)
                  rad = radal( a0, h1-h0 )
                  a1 = -sqrt( rad )
                  r1 = rp( r0, a1-a0 )
               end if
            else
               iquit = 1
            end if

! The ray has hit surface and is reflected.

            if( h1 .lt. 1.e-3 ) iquit = 1

            a0 = a1
```

```
         h0 = h1
         r0 = r1

      end do

      htfe(jr) = h0
      jr = jr + 1

   end do

   jr = jr - 1
   do i = jr, nrout
      htfe(i) = hmref
   end do

end if

end subroutine fillht
```

A.1.9 Subroutine GASABS

```
! *************************** SUBROUTINE GASABS *************************

! Module Name: GASABS

! Module Security Classification: UNCLASSIFIED

! PURPOSE: Computes sea-level gaseous absorption from temperature,
!          absolute humidity, and radio frequency using CCIR
!          Recommendation 676-1. This routine is good for frequencies
!          less than 57 GHz, air temperature from -20 to 40 degrees C,
!          and absolute humidity from 0 to 50 g/m3.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: ABSHUM, FREQ, TAIR

! OUTPUTS:
!   Argument List: NONE
!   Common: GASATT

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines called: NONE

! GLOSSARY:

!   Input Variables: See universal glossary for common variables.

!   Output Variables: See universal glossary for common variables.

!   Local Variables:
!       FGHZ = Frequency in GHz.
!       FGHZ2 = Square of frequency in GHz.
!       GAMMAO = oxygen absorption in dB/km.
!       GAMMAW = water vapor absorption in dB/km.

SUBROUTINE gasabs

use apm_mod
```

```
fghz = freq / 1.0E3
fghz2 = fghz * fghz

! Compute oxygen absorption for 15 degrees C air temperature.

t1 = 6.09 / (fghz2 + 0.227)
t2 = 4.81 / ((fghz - 57.0) ** 2 + 1.50)
gammao = (7.19E-3 + t1 + t2) * fghz2 * 1.0E-3

! Correct oxygen absorption for actual air temperature.

gammao = (1.0 - 0.01 * (Tair - 15.0)) * gammao

! Compute water vapor absorption.

t1 = 3.6  / ((fghz - 22.2) ** 2 + 8.5)
t2 = 10.6 / ((fghz - 183.3) ** 2 + 9.0)
t3 = 8.9  / ((fghz - 325.4) ** 2 + 26.3)
gammaw = (0.05 + 0.0021 * abshum +  t1 + t2 + t3) *     &
         fghz2 * abshum * 1.0E-4

! Compute total specific absorption for sea-level air in dB/km
! multiplied by conversion factor for computing loss in cB.

gasatt = (gammao + gammaw) * 1.e-2

END subroutine gasabs
```

A.1.10 Subroutine GETALN

```
! ************************** SUBROUTINE GETALN ******************************

! Module Name: GETALN

! Module Security Classification: UNCLASSIFIED

! Purpose:  Computes the impedance term ALPHAV and the complex index of
!           refraction for finite conductivity and vertical polarization
!           calculations. These formulas follow Kuttler's method. (Ref.
!           Kuttler's viewgraphs from PE modeler's workshop).

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: DELZ, DR, FKO, IG, IPOL, N
!   Public: CN2()

! OUTPUTS:
!   Argument List: NONE
!   Common: ALPHAV, C1X, C2X, RK, RT
!   Public: ROOT(), ROOTM()

! Modules Used: APM_MOD

! Calling Routines: PESTEP, XYINIT

! Routines called:
!   APM Specific: NONE
!   Intrinsic: CEXP, CLOG, CMPLX, CSQRT

! GLOSSARY:

!   Input Variables: See universal glossary for common variables.
```

```
!   Output Variables: See universal glossary for common variables.

!   Local Variables:
!       AD = Portion of exponent term for calculation of C1X and C2X.
!       RNG = complex refractive index.
!       S1 = Imaginary term in the formula for the square of the complex
!            index of refraction.

subroutine getaln

use apm_mod

complex  ad, sqrad, r2, a, rootln, qi, r2n, rng

data qi / (0., 1.) /        !Imaginary number i = sqrt(-1)

rng = csqrt( cn2(ig) )
alphav = qi * fko / rng    ! V pol
if( ipol .eq. 0 ) alphav = qi * fko * rng  ! H pol
ad = alphav * delz

sqrad = csqrt( 1. + ad**2 )

if( ipol .eq. 0 ) then
   rt = -sqrad - ad
else
   rt = sqrad - ad
end if
root(0) = cmplx( 1., 0. )
rootm(0) = root(0)
root(1) = rt
rootm(1) = -rt
do i = 2, n
   j = i-1
   root(i) = root(j) * rt
   rootm(i) = rootm(j) * (-rt)
end do

r2 = root(2)
r2n = root(n)**2
rk = 2.*(1. - r2) / (1. + r2) / (1. - r2n)
a = dr * qi / fko2
rootln = clog( rt )
ad = (rootln / delz)**2
c1x = cexp( a * ad )
ad = ( (rootln - qi * pi ) / delz )**2
c2x = cexp( a * ad )

end subroutine getaln
```

A.1.11 Subroutine GETMODE

```
!************************** SUBROUTINE GETMODE **********************

! Module Name: GETMODE

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine will determine what execution mode APM will run in,
!          based on environmental inputs.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
```

```
!   Common: ANTHT, FTER, ITPA, RMAX
!   Public: SLP(), TX()
!   Data: RTST

! OUTPUTS:
!   Argument List: RFLAT
!   Common: IHYBRID

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: ABS

! GLOSSARY: See universal glossary for common and public variables.

!   Input Variables: NONE

!   Output Variables:
!       RFLAT = Maximum range in meters at which the terrain profile
!               remains flat from the source.

!   Local Variables:
!       SLP_TOL = Terrain slope "fudge" factor.

subroutine getmode( RFLAT )

use apm_mod

data slptol / 1.e-3 /

rflat = rmax

if( antht .gt. 100. ) then
   ihybrid = 0           ! Use pure PE model
else
   ihybrid = 1        ! Use full hybrid mode

! Test to see if the first 2500 m. of terrain profile is flat.
! If not, then use partial hybrid mode (PE + XO).

   if( fter ) then
      i = 1
      do while(( abs( slp(i) ) .le. slptol ) .and. ( i .lt. itpa ))
         i = i + 1
      end do
      rflat = tx(i)
      if( rflat .le. rtst+.001 ) then
         ihybrid = 2
         rflat = rmax
      end if
   end if
end if

end subroutine getmode
```

A.1.12 Subroutine GETTHMAX

```
! **************************** SUBROUTINE GETTHMAX ***********************

! Module Name: GETTHMAX

! Module Security Classification: UNCLASSIFIED
```

```
! Purpose: This first part of this routine performs an iterative ray
!          trace such that, upon reflection, the ray clears the highest
!          terrain peak along its path by 20%.  Heights and angles of
!          this ray are stored at each output range. The 2nd part of this
!          routine determines the minimum PE propagation angle
!          necessary to meet the following criteria when using the full
!          hybrid mode (i.e,. IHYBRID=1): 1) Top of the PE
!          region must contain ALL trapping layers for all refractivity
!          profiles, 2) top of PE region must be at least 20% higher than
!          highest peak along terrain profile, 3) minimum PE propagation
!          angle must be at least as large as PSILIM.

! Version Number: 1.0

! INPUTS:
!   Argument List: ALFLIM, HTERMAX, HTEST, RFLAT, ZTEST
!   Common: ANTREF, FREQ, FTER, HTLIM, IHYBRID, IPOL, ISTART, ITPA,
!           LNMIN, NROUT, RMAX, WL, ZTOL
!   Data: RADC
!   Parameter: IRTEMP
!   Public: GR(), RNGOUT(), SLP(), TX(), TY(), ZRT()

! OUTPUTS:
!   Argument List: THETAMAX
!   Common: ALAUNCH, HTEMP(), IAP, PSILIM, RAYA(), RPEST, RTEMP(),
!           THETA75, ZMAX
!   Public: HLIM()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: FFTPAR
!   Intrinsic: ABS, AMAX0, AMAX1, AMIN0, AMIN1, EXP

! GLOSSARY: See universal glossary for common and data variables.

!   Input Variables:
!       ALFLIM = Elevation angle of RO limiting ray in radians. Used to
!                initialize launch angle in GETTHMAX routine.
!       HTERMAX = Maximum terrain height along profile path in meters.
!       HTEST = Minimum height in meters at which all trapping
!               refractivity features are below (includes some slop).
!       RFLAT = Maximum range in meters at which the terrain profile
!               remains flat from the source.
!       ZTEST = Height in PE region that must be reached for hybrid model.

!   Output Variables:
!       THETAMAX = Maximum propagation angle used in the PE model.

!   Local Variables:
!       A0 = Angle at start of trace in radians.
!       A1 = Angle at end of trace in radians.
!       AMLIM = Minimum PE angle limit, i.e., THETAMAX must be at least
!               this value.
!       AMXCUR = Maximum local angle along the traced ray up to ZLIM (with
!                minimum limit AMLIM).
!       ATEMP = Temporary THETAMAX, i.e., ATEMP = AMXCUR/.75
!       B1-3 = Coefficients of polynomial to determine AMLIM.
!       GRD = Gradient of current refractivity layer being traced through.
!       H0 = Height at start of trace in meters.
!       H1 = Height at end of trace in meters.
!       IQUIT = Integer flag indicating to quit (IQUIT=1) tracing of
!               current ray and begin again with a new launch angle.
!       IRAY = Integer flag to continue raytracing (IRAY = 0) or to stop
```

```
!              (IRAY = 1).
!        JL = Index for refractivity profile - current layer being traced
!             through.
!        KT = Counter index for terrain profile arrays TX() and TY().
!        R0 = Range at start of trace in meters.
!        R1 = Range at end of trace in meters.
!        SLOPE = Slope of current terrain segment.
!        T1-3 = Constants used in polynomial to determine AMLIM.
!        TOL = Iterative height tolerance to test if appropriate combination
!              of THETAMAX, FFT size, and ZMAX has been reached to meet
!              all criteria.
!        YN = Height of terrain at current range for traced ray.
!        YNT = Height of terrain at source.
!        ZLIMT = Maximum height to trace to (includes some slop).

subroutine getthmax( htest, htermax, rflat, ztest, alflim, THETAMAX )

use apm_mod

data t1, t2, t3 / 248.4, 2867., 2495. /
data b1, b2, b3 / 4.331, 1.420, .4091 /
data aoff / .37541 /                ! small angular offset

! Define in line ray trace functions:

radal( a, b ) = a**2 + 2. * grd * b                  !a=a0, b=h1-h0
rp( a, b ) = a + b / grd                             !a=r0, b=a1-a0
ap( a, b ) = a + b * grd                             !a=a0, b=r1-r0
hp( a, b, c ) = a + ( b**2 - c**2 ) / 2. / grd      !a=h0, b=a1, c=a0

! Determine minimum PE angle limit, AMLIM.

f1 = -freq / t1
f2 = -freq / t2
f3 = -freq / t3

amlim = aoff + b1 * exp(f1) + b2 * exp(f2) + b3 * exp(f3)
amlim = amin1( 4., amlim )
amlim = amlim * radc
aml = 0.
if( ipol .eq. 1 ) aml = 2. * amlim
if((fter) .and. ( ihybrid .eq. 1 )) then
   if( freq .le. 2000. ) aml = 1.5 * amlim
   if( freq .le. 1500. ) aml = 2. * amlim
   if( freq .le. 1000. ) aml = 2.5 * amlim
end if
amlim = amax1( amlim, aml )

alaunch = alflim
if( ihybrid .eq. 1 ) alaunch = -alaunch
zlimt = htlim - 1.e-3
iray = 0
yn = 0.
ynt = 0.
if(( ihybrid .eq. 1 ) .and. ( abs(ty(1)) .gt. 1.e-3 )) ynt = ty(1)
drtemp = rmax / float( irtemp )

! Begin iterative ray trace to determine the launch angle, and subsequently
! THETAMAX,

do while ( iray .eq. 0 )
   a0 = alaunch
   r0 = 0.
   h0 = antref
   jl = istart
   kt = 1
   slope = slp(kt)
```

```
   iquit = 0
   ro = 0.

! Begin tracing ray to each 1/IRTEMP range.

   do i = 1, irtemp

      ro = ro + drtemp

      do while( r0 .lt. ro )

         r1 = ro

         grd = gr(jl)
         if( a0 .lt. 0. ) grd = gr(jl-1)
         a1 = ap( a0, r1-r0 )

         if( sign(1.,a0) .ne. sign(1.,a1) ) then
            a1 = 0.
            r1 = rp( r0, a1-a0 )
         end if
         h1 = hp( h0, a1, a0 )

         if(( a1 .ge. 0. ) .and. ( h1 .ge. zrt(jl+1)-1.e-3 )) then
            h1 = zrt(jl+1)
            rad = radal( a0, h1-h0 )
            a1 = sqrt( rad )
            r1 = rp( r0, a1-a0 )
            jl = jl + 1
            h1 = amin1( htlim, zrt(jl) )
         elseif( a1 .le. 0. ) then
            if( h1 .le. ynt+1.e-3 ) then
               h1 = ynt
               rad = radal( a0, h1-h0 )
               a1 = -sqrt( rad )
               r1 = rp( r0, a1-a0 )
            end if
            if( h1 .le. zrt(jl-1)+1.e-3 ) then
               h1 = zrt(jl-1)
               rad = radal( a0, h1-h0 )
               a1 = -sqrt( rad )
               r1 = rp( r0, a1-a0 )
               jl = amax0( 0, jl - 1 )
            end if
         end if

! The ray has hit surface and is reflected.

         if( h1 .lt. ynt+1.e-3 ) then
            a1 = -a1
            psilim = a1
            rpest = r1
           if( r1 .gt. rflat ) iquit = 1
         end if

         h0 = h1
         r0 = r1
         a0 = a1
         if( iquit .eq. 1 ) exit
      end do

! Check to see that current height of ray is at least 20% higher than
! current terrain height.

      if( fter ) then
         do while((r0 .gt. tx(kt+1)) .and. (kt .lt. itpa))
```

```
            kt = kt + 1
            slope = slp(kt)
         end do
         yn = 1.2 * (ty(kt) + slope * ( r0 - tx(kt) ))
      end if

      raya(i) = a0
      htemp(i) = h0
      rtemp(i) = r0

      if( ihybrid .eq. 1 ) then
         if(( h0 .lt. yn ) .and. (r0 .gt. rflat)) iquit = 1
      else
         if( h0 .lt. yn ) iquit = 1
      end if
      if( h0 .ge. zlimt ) exit
      if( iquit .eq. 1 ) exit

   end do

! If H0 is less than current terrain height then increase (steepen) angle
! and perform ray trace again. Angle is initially downgoing for IHYBRID=1
! and upgoing otherwise.

   if( iquit .eq. 1 ) then
      if( ihybrid .eq. 1 ) then
         alaunch = alaunch - 1.e-3
      else
         alaunch = alaunch + 1.e-3
      end if
   else

! If ZLIM is reached with no problems, then set initial launch angle -
! ray has been found with all heights, ranges, and angles stored.  Set flag
! to quit.

      iray = 1
      ihmax = i
   end if
end do

do i = ihmax, irtemp
   htemp(i) = h0
   raya(i) = a0
   rtemp(i) = rmax
end do
ihmax = amin0( ihmax, irtemp )

! Determine at what index the local ray angles become positive,
! i.e., after reflection.

j = 1
do while( raya(j) .lt. 0. )
   j = j + 1
end do
iap = j

! Now determine THETAMAX for PE region based on local angles along ray.

iok = 0
iflag = 0
zlim = 0.
amxcur = 0.

do while( iok .eq. 0 )
   j = iap
   zt = ztest - 1.e-3
```

```
   do j = iap, ihmax
      if( htemp(j) .gt. zt ) exit
   end do
   ist = amin0( j, ihmax )

   amxcur = abs( raya(1) )
   do i = 2, ist
      if( abs( raya(i) ) .gt. amxcur ) amxcur = raya(i)
   end do

   amxcur = amax1( amlim, amxcur )
   atemp = amxcur / .75

   if( ihybrid .eq. 2 ) ztest = amax1( antref, htest, 1.2*htermax, 1000. )

! Compute new ZTEST, ZMAX, DELZ, LN, N.

   call fftpar( lnmin, wl, atemp, iflag, ZTEST, ZMAX, DELZ, LN, N )

   if( iflag .eq. 0 ) then
      iflag = 1
      if( ihybrid .ne. 1 ) iok = 1
   elseif(( iflag .eq. 1 ) .and. ( ihybrid .ne. 2)) then
      tol = abs( ztest - zlim ) / ztest
      if( tol .le. ztol ) iok = 1
   end if
   zlim = ztest

end do

theta75 = amxcur
thetamax = atemp

!Before exiting, fill in array HLIM().

if( ihybrid .eq. 1 ) then
   a0 = psilim
   r0 = rpest
   h0 = 0.
   jl = 0
else
   a0 = alaunch
   r0 = 0.
   h0 = antref
   jl = istart
end if
do i = 1, nrout
   if( rngout(i) .lt. rpest ) then
      hlim(i) = 0.
   else
      ro = rngout(i)
      do while(( r0 .lt. ro ) .and. ( h0 .le. htlim ))
         r1 = ro
         grd = gr(jl)
         if( a0 .lt. 0. ) grd = gr(jl-1)
         a1 = ap( a0, r1-r0 )
         if( sign(1.,a0) .ne. sign(1.,a1) ) then
            a1 = 0.
            r1 = rp( r0, a1-a0 )
         end if
         h1 = hp( h0, a1, a0 )
         if( h1 .ge. zrt(jl+1)-1.e-3 ) then
            h1 = zrt(jl+1)
            rad = radal( a0, h1-h0 )
            a1 = sqrt( rad )
            r1 = rp( r0, a1-a0 )
            jl = jl + 1
```

```
         end if
         h0 = h1
         r0 = r1
         a0 = a1
      end do

      hlim(i) = h0
   end if
end do

end subroutine getthmax
```

A.1.13 Subroutine INTPROF

```
! ************************** SUBROUTINE INTPROF ******************************

! Module Name: INTPROF

! Module Security Classification: UNCLASSIFIED

! Purpose:  Performs a linear interpolation vertically with height on the
!           refractivity profile.  Stores interpolated profile in PROFINT().

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: CON, N, NLVL
!   Public: HREF(), HT(), REFREF()

! OUTPUTS:
!   Argument List: NONE
!   Common: NONE
!   Public: PROFINT()

! Modules Used: APM_MOD

! Calling Routines: APMINIT, REFINTER

! Routines Called: NONE

! GLOSSARY:

!   Input Variables: See universal glossary for common variables.

!   Output Variables: See universal glossary for common variables.

!   Local Variables:
!       HEIGHT = Height to interpolate to.
!       FRAC = Fractional height for interpolation.


SUBROUTINE intprof

use apm_mod

j = 1
k = 0

profint(0) = refref(0) * con
DO I = 1, N

   height = ht(i)
   do while(( height .gt. href(j) ) .and. ( j .lt. nlvl ))
      j = j + 1
```

```
      k = j - 1
   end do
   FRAC = (height - href(k)) / (href(J) - href(k))
   profint(I) = (refref(k) + FRAC * (refref(J) - refref(k))) * con
end do

END subroutine intprof
```

A.1.14 Subroutine PHASE1

```
! ***************************** SUBROUTINE PHASE1 ****************************

! Module Name: PHASE1

! Module Security Classification: UNCLASSIFIED

! Purpose: Initialize free-space propagator array FRSP() using wide-angle
!          propagator.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: CNST, DR, FKO, FNORM, N, N34
!   Public: FILT()

! OUTPUTS:
!   Argument List: NONE
!   Common: NONE
!   Public: FRSP()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: AMIN1, CMPLX, COS, FLOAT, SIN, SQRT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables: NONE

!   Output Variables: NONE

!   Local Variables:
!       ANG = Exponent term:
!             ANG = -i*dr*k*[1-sqrt(1-(p/k)**2)] where k is the free-space
!             wavenumber, p is the transform variable (p=k*sin(theta)), and
!             i is the imaginary number (i=sqrt(-1)).

SUBROUTINE PHASE1

use apm_mod

double precision cak

drfk = dr * fko
DO I=0,N
   ak = float(i) * cnst
   aksq=ak * ak
   aksq = amin1( 1., aksq )
   cak = sqrt(1. - aksq)
   ang = drfk * ( 1.d0 - cak )
   ca = cos( ang )
```

```
   sa = -sin( ang )
   frsp(i) = fnorm * cmplx( ca, sa )
end do

! Filter the upper 1/4 of the propagator arrays.

do i = n34, n
   attn = filt(i-n34)
   frsp(i) = attn * frsp(i)
end do

END subroutine phase1
```

A.1.15 Subroutine PHASE2

```
! ************************** SUBROUTINE PHASE2 ******************************

! Module Name: PHASE2

! Module Security Classification: UNCLASSIFIED

! Purpose: Calculates the environmental phase term for a given profile, then
!          stores in array ENVPR().

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: DR, N, N34
!   Public: FILT(), PROFINT()

! OUTPUTS:
!   Argument List: NONE
!   Common: NONE
!   Public: ENVPR()

! Calling Routines: APMINIT, PESTEP

! Routines called:
!   APM Specific: NONE
!   Intrinsic: CMPLX, COS, SIN

! GLOSSARY:

!   Input Variables: NONE

!   Output Variables: NONE

!   Local Variables:
!       ANG = Exponent term: ANG = i*dr*k*1e-6*M(z) where, i is the
!             imaginary number (i=sqrt(-1)), k is the free-space wavenumber,
!             and M(z) is the modified refractivity.

SUBROUTINE PHASE2

use apm_mod

do i = 0, n
   ang = dr * profint(i)
   ca = cos( ang )
   sa = sin( ang )
   envpr(i) = cmplx( ca, sa )
end do

! Filter upper 1/4 of the arrays.
```

```
do i = n34, n
   attn = filt(i-n34)
   envpr(i) = attn * envpr(i)
end do

END subroutine phase2
```

A.1.16 Subroutine PROFREF

```
! ****************************** SUBROUTINE PROFREF *************************

! Module Name: PROFREF

! Module Security Classification: UNCLASSIFIED

! Purpose: This subroutine determines the refractivity profile with respect
!          to the reference height YREF which, depending on the value of
!          IFLAG, can be HMINTER or the local ground height above HMINTER.

! Version Number: 1.0

! INPUTS:
!   Argument List: IFLAG, YREF
!   Common: IEXTRA, LVLEP
!   Public: HTDUM(), REFDUM()

! OUTPUTS:
!   Argument List: NONE
!   Common: LVLEP, NLVL
!   Public: HREF(), HTDUM(), REFDUM(), REFREF()

! Modules Used: APM_MOD

! Calling Routines: REFINIT, REFINTER

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: ABS,INT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       IFLAG = 0: Profile arrays REFREF() and HREF() will be referenced to
!                  height HMINTER, and will also be used to initialize
!                  REFDUM() and HTDUM().
!             = 1: Profile arrays REFREF() and HREF() will be referenced to
!                  local ground height.
!       YREF = Reference height in meters at current range.

!   Output Variables:NONE

!   Local Variables:
!       FRAC = Fractional height used for interpolation
!       IBMSL = Integer flag indicating if YREF is below mean sea level (msl)
!               IBMSL = 0 -> YREF not below msl
!               IBMSL = 1 -> YREF below msl
!       JS = Integer index indicating at what index/level in array HTDUM()
!            YREF is located.
!       RMU = Interpolated M-unit value at height YREF.
!       NEWL = New/adjusted number of levels to be stored in HREF() and
!              REFREF().

subroutine profref( yref, iflag )
```

```
use apm_mod

nlvl = lvlep
href = 0.          !array
refref = 0.        !array

if( abs(yref) .gt. 1.e-3 ) then

   ibmsl = 0
   js = -1

! Check to see if reference height is below mean sea level.

   if( yref .lt. 0. ) then
      ibmsl = 1
      js = 0

! Get refractivity profile level at which the height of the ground is just
! above. This level is JS.

   else
      nlvlm1 = nlvl - 1
      do i = 0, nlvlm1
         if(( yref .le. htdum(i+1) ) .and. ( yref .gt. htdum(i) )) js = i
      end do
   end if

! Determine the refractivity value at the ground and fill arrays HREF() and
! REFREF() with refractivity profile where height 0. now refers to the ground
! reference,i.e., either local ground height or HMINTER.

   if(( js .gt. -1 ) .or. ( ibmsl .eq. 1 )) then
      jsp1 = js + 1
      frac = (yref - htdum(js)) / (htdum(jsp1) - htdum(js))
      rmu = refdum(js) + frac * (refdum(jsp1) - refdum(js))
      if((iextra .eq. 0) .and. (ibmsl .eq. 1)) rmu = refdum(js) + frac * .118
      if( int( frac ) .eq. 1 ) js = jsp1
      newl = nlvl - js
      refref(0) = rmu
      href(0) = 0.
      k = js + 1
      do jk = 1, newl
         refref(jk) = refdum(k)
         href(jk) = htdum(k) - yref
         k = k + 1
      end do
      nlvl = newl
      if( iflag .eq. 0 ) then
         lvlep = nlvl
         refdum = refref         !array
         htdum = href            !array
      end if
   end if
else

! If the reference height is 0. then HREF() and REFREF() are equal.

   do i = 0, nlvl
      href(i) = htdum(i)
      refref(i) = refdum(i)
   end do
end if

end subroutine profref
```

A.1.17 Subroutine REFINIT

```
! ************************** SUBROUTINE REFINIT ******************************

! Module Name: REFINIT

! Module Security Classification: UNCLASSIFIED

! Purpose: Initializes refractivity arrays used for subsequent PE and RO
!          calculations.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: ANTREF, HMINTER, LERR12, LVLP, NPROF, RMAX
!   Public: HMSL(,), REFMSL(,), RNGPROF()

! OUTPUTS:
!   Argument List: HTRAP, HTHICK, IERROR, RMMIN, RMMAX
!   Common: IS, ISTART, LEVELS, LVLEP, LVLP, NLVL, RV2
!   Public: HMSL(,), HTDUM(), GR(), Q(), REFDUM(), REFMSL(,), RM(), ZRT()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: PROFREF, REMDUP
!   Intrinsic: ABS, SIGN

! GLOSSARY: See universal glossary for common variables

!   Input Variables: NONE

!   Output Variables:

!       HTHICK = Thickness in meters of highest trapping layer from all
!                refractivity profiles.
!       HTRAP = Height of highest trapping layer in meters from all
!               refractivity profiles.
!       IERROR =  -6 : Last range in terrain profile is less than RMAX.
!                      (Will only return this error if error flag LERR6
!                      is set to .TRUE.).
!              = -12 : Range of last refractivity profile entered (for
!                      range dependent case) is less than RMAX. (This is
!                      returned from subroutine REFINIT).  Will only
!                      return this error if error flag LERR12 is set to
!                      .TRUE.).
!              = -13 : Height of first level in any user-specified refrac-
!                      tivity profile is greater than 0.  First height must
!                      be at m.s.l. (0.) or >0. if below m.s.l.
!              = -14 : Last gradient in user-provided refractivity profile
!                      is negative.
!       RMMAX = Maximum M-unit value (x10e-6) of refractivity profile at
!               range 0.
!       RMMIN = Minimum M-unit value (x10e-6) of refractivity profile at
!               range 0.

!   Local Variables:

!       GRD = Gradient at current height/refractivity level.
!       HLARGE = Maximum height limit for last level in height/refractivity
!                profiles.
!       ZHI = Height at next higher profile point in meters.
!       ZLO = Height at next lower profile point in meters.
```

```
subroutine refinit( HTRAP, HTHICK, RMMIN, RMMAX, IERROR )

use apm_mod

data hlarge/ 1.e6 /

ierror = 0

! Test to see if last profile entered ( for range dependent case ) meets or
! exceeds RMAX, otherwise, return error (unless error trapping is turned off -
! LERR12 = .FALSE.).

if( nprof .gt. 1 ) then
   if(( rngprof(nprof) .lt. rmax ) .and. ( lerr12 )) then
      ierror = -12
      return
   end if
end if

do i = 1,nprof

! Test to see that every user-specified profile begins with 0. height for
! 1st level in profile (allow for <0. height if below m.s.l.).

   if( hmsl(0,i) .gt. 0. ) then
      ierror = -13
      return
   end if

   hdif = 0.
   lvlm1 = lvlp
   lvlm2 = lvlp
   do while( hdif .le. 1.e-6 )
      lvlm1 = lvlm1 - 1
      lvlm2 = lvlm1 - 1
      hdif = hmsl(lvlm1,i) - hmsl(lvlm2,i)
   end do
   grd = (refmsl(lvlm1,i)-refmsl(lvlm2,i)) / hdif

! If last gradient in refractivity profile is negative then return error.

   if( grd .lt. 0 ) then
      ierror = -14
      return
   end if

! Add extra level to tabulated profiles with extrapolated gradient. Test on
! HDIF greater than 0 for profiles that contain multiple height/M-unit values
! that are equal. LVLP is already one more than # of actual levels in profiles.

   hmsl(lvlp, i) = hlarge
   refmsl(lvlp, i) = (hlarge-hmsl(lvlm1,i)) * grd + refmsl( lvlm1, i )
end do

is = 1
rv2=rngprof(is)

do i = 0, lvlp
   refdum(i) = refmsl( i, is )
   htdum(i) = hmsl( i, is )
end do

lvlep = lvlp

! Remove any duplicate levels in first profile and adjust HTDUM() and
! REFDUM() to minimum terrain height.
```

```
call remdup
call profref( hminter, 0 )

! NLVL is now the number of height/refractivity levels in adjusted HTDUM()
! and REFDUM().
! Find height and thickness of highest trapping layer, if one exists,
! relative to HMINTER.

htrap = 0.
hthick = 0.
do i = 1, nprof
   do j = 0, lvlp-1
      grd = refmsl(j+1,i) - refmsl(j,i)
      hp1 = hmsl(j+1,i) - hminter
      if(( grd .lt. 0. ) .and. (hp1 .gt. htrap)) then
         htrap = hp1
         hp0 = hmsl(j,i) - hminter
         hthick = hp1 - hp0
      end if
   end do
end do

! Build Z and RM arrays for RO calculations if needed. Add level for
! ANTHT, if needed.

zrt(0) = htdum(0)
rm(0) = 1.e-6 * refdum(0)
i = 0
istart = 0

DO j = 1, nlvl
   zhi = htdum(j)
   zlo = htdum(j-1)
   i = i + 1
   IF (ABS(zhi - antref) .LT. 1.E-3) istart = i
   IF ((istart .EQ. 0) .AND. (zhi .GT. antref)) THEN
      zrt(i) = antref
      ip1 = i + 1
      im1 = i - 1
      zrt(ip1) = zhi
      rm(ip1) = 1.e-6 * refdum(j)
      drmdz = (rm(ip1) - rm(im1)) / (zrt(ip1) - zrt(im1))
      rm(i) = rm(im1) + drmdz * (antref - zrt(im1))
      istart = i
      i = i + 1
   ELSE
      zrt(i) = zhi
      rm(i) = 1.e-6 * refdum(j)
   END IF
END DO

! Highest profile point exceeds antenna height.  Total number of
! points in z array reduced by 1 since highest level is not needed.

levels = i - 1

! Build GR and Q arrays for ray-optics and ray-tracing routines.

do i = 0, levels
   ip1 = i + 1
   rmd = rm(ip1) - rm(i)
   grd = rmd / (zrt(ip1) - zrt(i))
   if( abs( grd ) .lt. 1.e-8 ) grd = sign( 1., grd )*1.e-8
   gr(i) = grd
   q(i) = 2. * rmd
end do
```

```
! Determine minimum RM (1.E-6*M) on profile, and maximum RM at or
! below the transmitter.

rmmin = rm(0)
rmmax = rmmin
DO i = 1, nlvl
   rmmin = amin1( rm(i), rmmin )
   IF((rm(i) .GT. rmmax) .AND. ( i .LE. istart )) rmmax = rm(i)
END DO

end subroutine refinit
```

A.1.18 Subroutine SINFFT

```
SUBROUTINE SINFFT ( N, X )
!
!**********************************************************************
!*                                                                    *
!*                                                                    *
!* PURPOSE:    SINFFT replaces the real array X()                     *
!*             by its finite discrete sine transform                  *
!*                                                                    *
!* METHOD :                                                           *
!*                                                                    *
!*     The algorithm is based on a mixed radix (8-4-2) real vector    *
!*     fast Fourier synthesis routine published by Bergland:          *
!*                                                                    *
!*     ( G.D. Bergland, 'A Radix-eight Fast Fourier Transform         *
!*      Subroutine for Real-valued Series,' IEEE Transactions on      *
!*      Audio and Electro-acoustics', vol. AU-17, pp. 138-144, 1969 ) *
!*                                                                    *
!*     and sine and cosine transform algorithms for real series       *
!*     published by Cooley, Lewis, and Welch:                         *
!*                                                                    *
!*     (J.W. COOLEY, P.A.W. LEWIS AND P.D. WELSH, 'The Fast Fourier   *
!*     Transform Algorithm: Programming Considerations in the         *
!*     Calculation of Sine, Cosine and Laplace Transforms',           *
!*     J. SOUND VIB., vol. 12, pp. 315-337, 1970 ).                   *
!*                                                                    *
!*                                                                    *
!* ARGUMENTS:                                                         *
!*                        -- INPUT --                                 *
!*                                                                    *
!*      N...... transform size ( = 2**N )                             *
!                                                                    *
!     X().... data array dimensioned 2**N in calling program         *
!                                                                    *
!                       -- OUTPUT --                                 *
!                                                                    *
!     X().... sine transform                                         *
!                                                                    *
! TABLES:  array     required size                                   *
!                                                                    *
!  .           B          2**N                                       *
!              JINDX      2**(N-1)                                   *
!              COSTBL     2**(N-4)                                   *
!              SINTBL     2**(N-4)                                   *
!                                                                    *
! Sub-programs called:  -                                            *
!                                                                    *
!                R8SYN..... (radix 8 synthesis)                      *
!                                                                    *
!*********************************************************************
```

```
      INTEGER*4    N

      DIMENSION    X(0:*)
      INTEGER*4    NMAX2, NMAX16, NP, NPD2, NPD4

      real, allocatable :: b(:), sines(:), costbl(:), sintbl(:)
      integer*4, allocatable :: jindx(:)

      SAVE B, COSTBL, JINDX, SINES, SINTBL, nmax2, nmax16
      SAVE NSAVE, N4, N8, NP, NPD2, NPD4, NPD16, NPM1

      DOUBLE PRECISION ARG, DT, dpi
      DATA NSAVE  / 0 /
      DATA dpi     / 3.1415926535897932D0 /

!-------------------------------------------------------------------+

      IF(( N .NE. NSAVE ) .and. ( n .gt. 0 )) THEN

        NP = 2**N
        nmax2 = np / 2
        nmax16 = np / 16

        if( allocated ( b ) ) deallocate ( b )
        allocate ( b(np) )
        b = 0.

        if( allocated ( jindx ) ) deallocate ( jindx )
        allocate ( jindx(nmax2) )
        jindx = 0

        if( allocated ( sines ) ) deallocate ( sines )
        allocate ( sines(np) )
        sines = 0.

        if( allocated ( costbl ) ) deallocate ( costbl )
        allocate ( costbl(nmax16) )
        costbl = 0.

        if( allocated ( sintbl ) ) deallocate ( sintbl )
        allocate ( sintbl(nmax16) )
        sintbl = 0.
                                       !compute constants and construct tables
        NSAVE = N
        N8 = NSAVE / 3
        N4 = NSAVE - 3 * N8 - 1
        NPD2  = NP / 2
        NPD4  = NP / 4
        NPD16 = NP / 16
        NPM1  = NP - 1
                                            ! build reciprical sine table
        DT    = dpi / FLOAT ( NP )
        DO J = 1, NPM1
           ARG = DT * J
           SINES ( J ) = (0.5D0 / SIN ( ARG ))
        end do
                                        !construct bit reversed subscript table
        J1 = 0
        DO J = 1, NPD2 - 1
           J2 = NPD2
           do while( IAND ( J1, J2 ) .NE. 0 )
              J1 = IABS( J1 - J2 )
              J2 = J2 / 2
           end do
           J1 = J1 + J2
           JINDX ( J ) = J1
        end do
```

```
                          !form the trig tables for the radix-8 passes;
                          !tables are stored in bit reversed order.
      J1 = 0
      DO J = 1, NPD16 - 1
         J2 = NPD16
         do while ( IAND ( J1, J2 ) .NE. 0 )
            J1 = IABS( J1 - J2 )
            J2 = J2 / 2
         end do
         J1  = J1 + J2
         ARG = DT * FLOAT (J1)
         COSTBL ( J ) =  COS ( ARG )
         SINTBL ( J ) = -SIN ( ARG )
      end do

   elseif( n .eq. -1 ) then

!End of APM run - deallocate arrays and return to main driver program.


      if( allocated( b ) ) deallocate( b, stat = ierror )
      if( allocated( sines ) )deallocate( sines, stat = ierror )
      if( allocated( costbl ) )deallocate( costbl, stat = ierror )
      if( allocated( sintbl ) )deallocate( sintbl, stat = ierror )
      if( allocated( jindx ) )deallocate( jindx, stat = ierror )
      nsave = 0
      return

   ENDIF


!***  form the input Fourier coefficients ***

!sine transform

   B ( 1 ) = -2. * X ( 1 )
   B ( 2 ) =  2. * X ( NPM1 )
   J1 = 0
   DO J = 3, NPM1, 2
      J1 = J1 + 1
      J2 = JINDX ( J1 )
      B ( J )      = X ( J2 - 1 ) - X ( J2 + 1 )
      B ( J + 1 ) = X ( NP-J2 )
   end do

!                   ************************************
!                   *                                  *
!                   *    Begin Fast Fourier Synthesis  *
!                   *                                  *
!                   ************************************
!
   IF ( N8 .NE. 0 ) THEN
!                                                   radix-8 iterations
      INTT = 1
      NT = NPD16
      DO J = 1, N8
         J1 =  1 + INTT
         J2 = J1 + INTT
         J3 = J2 + INTT
         J4 = J3 + INTT
         J5 = J4 + INTT
         J6 = J5 + INTT
         J7 = J6 + INTT
!**
         CALL R8SYN (INTT, NT, COSTBL, SINTBL, B(1), B(J1), B(J2), &
                     B(J3), B(J4), B(J5), B(J6), B(J7)  )
```

```
!**
         NT = NT / 8
         INTT = 8 * INTT
      end do

   ENDIF

!                                                          radix-4 iteration
   IF ( N4 .GT. 0 ) THEN
      J1 = NPD4
      J2 = 2*NPD4
      J3 = 3*NPD4
      DO J = 1, NPD4
         T0 = B(J) + B(J + J1)
         T1 = B(J) - B(J + J1)
         T2 = 2. * B(J + J2)
         T3 = 2. * B(J + J3)
         B(J)      = T0 + T2
         B(J + J2) = T0 - T2
         B(J + J1) = T1 + T3
         B(J + J3) = T1 - T3
      end do

   ELSE IF ( N4 .EQ. 0 ) THEN
!                                                          radix-2 iteration
      K = NPD2
      DO J = 1, NPD2
         K    = K + 1
         T    = B(J) + B (K)
         B(K) = B(J) - B (K)
         B(J) = T
      end do
   ENDIF

!                    ***********************
!                    *                     *
!                    *    Form Transform   *
!                    *                     *
!                    ***********************
!
!
!                                                          sine transform
   J1 = NP
   DO J = 1, NPM1
      X(J) = .25*(( B(J+1) + B(J1)) * SINES(J) - B(J+1) + B(J1))
      J1 = J1 - 1
   end do

   END subroutine sinfft

   SUBROUTINE R8SYN ( INTT, NT, COSTBL, SINTBL, B0, B1, B2, B3,&
                      B4, B5, B6, B7 )

!**********************************************************************

! PURPOSE:    Radix-8 synthesis subroutine used by mixed radix driver.

!**********************************************************************

   DIMENSION   COSTBL(*), SINTBL(*)
   DIMENSION B0(*), B1(*), B2(*), B3(*), B4(*), B5(*), B6(*), B7(*)

!           ///     Local variables     ///

   DOUBLE PRECISION C1, C2, C3, C4, C5, C6, C7
   DOUBLE PRECISION S1, S2, S3, S4, S5, S6, S7
   DOUBLE PRECISION CPI4, CPI8, R2, SPI8
```

```
      SAVE  CPI4, CPI8,  R2, SPI8

      DATA  R2    / 1.41421356237310D+0 /, &
            CPI4  / 0.70710678118655D+0 /, &
            CPI8  / 0.92387953251129D+0 /, &
            SPI8  / 0.38268343236509D+0 /

!---------------------------------------------------------------------+

      JT = 0
      JL = 2
      JR = 2
      JI = 3
      INT8 = 8 * INTT

      DO K = 1, INTT
         T0 = B0(K) + B1(K)
         T1 = B0(K) - B1(K)
         T2 = B2(K) + B2(K)
         T3 = B3(K) + B3(K)
         T4 = B4(K) + B6(K)
         T5 = B4(K) - B6(K)
         T6 = B7(K) - B5(K)
         T7 = B7(K) + B5(K)
         T8 = R2 * (T7 - T5)
         T5 = R2 * (T7 + T5)
         TT0 = T0 + T2
         T2  = T0 - T2
         TT1 = T1 + T3
         T3  = T1 - T3
         T4  = T4 + T4
         T6  = T6 + T6

         B0(K) = TT0 + T4
         B4(K) = TT0 - T4
         B1(K) = TT1 + T5
         B5(K) = TT1 - T5
         B2(K) = T2 + T6
         B6(K) = T2 - T6
         B3(K) = T3 + T8
         B7(K) = T3 - T8
      end do

      IF ( NT .EQ. 0 )                    RETURN

      K0 = INT8 + 1
      KLAST = INT8 + INTT

      DO K = K0, KLAST
         T1 = B0(K) + B6(K)
         T3 = B0(K) - B6(K)
         T2 = B7(K) - B1(K)
         T4 = B7(K) + B1(K)
         T5 = B2(K) + B4(K)
         T7 = B2(K) - B4(K)
         T6 = B5(K) - B3(K)
         T8 = B5(K) + B3(K)

         B0(K) = (T1 + T5) + (T1 + T5)
         B4(K) = (T2 + T6) + (T2 + T6)
         T5    = T1 - T5
         T6    = T2 - T6
         B2(K) = R2 * (T6 + T5)
         B6(K) = R2 * (T6 - T5)
         T1    = T3 * CPI8 + T4 * SPI8
         T2    = T4 * CPI8 - T3 * SPI8
```

```
      T3    = T8 * CPI8 - T7 * SPI8
      T4    =  - T7 * CPI8 - T8 * SPI8
      B1(K) = (T1 + T3) + (T1 + T3)
      B5(K) = (T2 + T4) + (T2 + T4)
      T3    = T1 - T3
      T4    = T2 - T4
      B3(K) = R2 * (T4 + T3)
      B7(K) = R2 * (T4 - T3)
   end do

   DO JT = 1, NT-1
      C1 = COSTBL(JT)
      S1 = SINTBL(JT)
      C2 = C1 * C1 - S1 * S1
      S2 = C1 * S1 + C1 * S1
      C3 = C1 * C2 - S1 * S2
      S3 = C2 * S1 + S2 * C1
      C4 = C2 * C2 - S2 * S2
      S4 = C2 * S2 + C2 * S2
      C5 = C2 * C3 - S2 * S3
      S5 = C3 * S2 + S3 * C2
      C6 = C3 * C3 - S3 * S3
      S6 = C3 * S3 + C3 * S3
      C7 = C3 * C4 - S3 * S4
      S7 = C4 * S3 + S4 * C3

      K  = JI * INT8
      J0 = JR * INT8 + 1
      JLAST = J0 + INTT - 1

      DO J = J0, JLAST

         K   = K + 1
         TR0 = B0(J) + B6(K)
         TR1 = B0(J) - B6(K)
         TI0 = B7(K) - B1(J)
         TI1 = B7(K) + B1(J)
         TR2 = B4(K) + B2(J)
         TI3 = B4(K) - B2(J)
         TI2 = B5(K) - B3(J)
         TR3 = B5(K) + B3(J)
         TR4 = B4(J) + B2(K)
         T0  = B4(J) - B2(K)
         TI4 = B3(K) - B5(J)
         T1  = B3(K) + B5(J)
         TR5 = CPI4 * (T1 + T0)
         TI5 = CPI4 * (T1 - T0)
         TR6 = B6(J) + B0(K)
         T0  = B6(J) - B0(K)
         TI6 = B1(K) - B7(J)
         T1  = B1(K) + B7(J)
         TR7  =  - CPI4 * (T0 - T1)
         TI7  =  - CPI4 * (T0 + T1)
         T0   = TR0 + TR2
         TR2  = TR0 - TR2
         T1   = TI0 + TI2
         TI2  = TI0 - TI2
         T2   = TR1 + TR3
         TR3  = TR1 - TR3
         T3   = TI1 + TI3
         TI3  = TI1 - TI3
         T5   = TI4 + TI6
         TTR6 = TI4 - TI6
         TI6  = TR6 - TR4
         T4   = TR4 + TR6
         T7   = TI5 + TI7
         TTR7 = TI5 - TI7
```

```
         TI7   = TR7 - TR5
         T6    = TR5 + TR7

         B0(J) = T0 + T4
         B0(K) = T1 + T5
         B4(J) = C4 * (T0 - T4)      - S4 * (T1 - T5)
         B4(K) = C4 * (T1 - T5)      + S4 * (T0 - T4)

         B1(J) = C1 * (T2 + T6)      - S1 * (T3 + T7)
         B1(K) = C1 * (T3 + T7)      + S1 * (T2 + T6)
         B5(J) = C5 * (T2 - T6)      - S5 * (T3 - T7)
         B5(K) = C5 * (T3 - T7)      + S5 * (T2 - T6)

         B2(J) = C2 * (TR2 + TTR6)  - S2 * (TI2 + TI6)
         B2(K) = C2 * (TI2 + TI6)   + S2 * (TR2 + TTR6)
         B6(J) = C6 * (TR2 - TTR6)  - S6 * (TI2 - TI6)
         B6(K) = C6 * (TI2 - TI6)   + S6 * (TR2 - TTR6)

         B3(J) = C3 * (TR3 + TTR7)  - S3 * (TI3 + TI7)
         B3(K) = C3 * (TI3 + TI7)   + S3 * (TR3 + TTR7)
         B7(J) = C7 * (TR3 - TTR7)  - S7 * (TI3 - TI7)
         B7(K) = C7 * (TI3 - TI7)   + S7 * (TR3 - TTR7)

      end do

      JR = JR + 2
      JI = JI - 2
      IF ( JI .GT. JL) cycle
      JI = JR + JR - 1
      JL = JR

   end do

   END subroutine r8syn
```

A.1.19 Subroutine TERINIT

```
!************************* SUBROUTINE TERINIT *************************

! Module Name: TERINIT

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine initializes the arrays TX() and TY() and all
!          associated terrain variables.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: ANTHT, HMAX, HMIN, ITP, ITPA, RMAX
!   Public: TERX(), TERY()

! OUTPUTS:
!   Argument List: ANGU, HTERMAX, IERROR, RFIX
!   Common: ANTREF, FTER, HMINTER, HMREF, HTLIM
!   Public: SLP(), TX(), TY()

! Modules Used: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: AMAX1, ATAN, FLOAT, NINT
```

```
! GLOSSARY: See universal glossary for common variables and parameters.

!   Input Variables: NONE

!   Output Variables:
!       ANGU = Maximum tangent ray angle from source to terrain peak
!              along profile path.
!       HTERMAX = Maximum terrain height along profile path in meters.
!       IERROR = Integer value that is returned if any errors exist in input
!                data:
!                -6 : Last range in terrain profile is less than RMAX.
!                     (Will only return this error if error flag LERR6
!                     is set to .TRUE.).
!                -8 : HMAX is less than maximum height of terrain profile.
!                -9 : Antenna height w.r.t. msl is greater than maximum
!                     height HMAX.
!               -17 : Range points of terrain profile are not increasing.
!               -18 : First range point is not 0.
!       RFIX = If terrain profile points are equally spaced, this is
!              automatically determined and range spacing is set to RFIX,
!              otherwise, RFIX = 0.

!   Local Variables:
!       ANGLE = Tangent angle from source to each terrain point in radians
!       HDEG = 1/2 degree in radians.
!       RDIF1 = Difference between adjacent terrain point elevations.
!       RDIF2 = Difference between next adjacent terrain point elevations.
!       RDIFSUM = Running sum of adjacent terrain point differences.
!       RFRAC = Maximum fraction between adjacent terrain point differences.
!       SLOPE = Slope of terrain segment.
!       X1, X2 = Range of Ith and I+1 terrain point, respectively.
!       Y1, Y2 = Height of Ith and I+1 terrain point, respectively.
!       XDIF = Range difference between adjacent terrain points.
!       YDIF = Height difference between adjacent terrain points.

subroutine terinit( ANGU, RFIX, HTERMAX, IERROR )

use apm_mod

data hdeg / 8.726646e-3 /     ! 1/2 degree

fter = .false.
ierror = 0
angu = 0.
hminter = 0.
antref = antht
htermax = 0.

if( itp .gt. 0 ) fter = .true.

! Check that all terrain range points are increasing.

if( fter ) then
   do i = 1, itp-1
      ip1 = i + 1
      if( terx(ip1) .lt. terx(i) ) then
         ierror = -17
         return
      end if
   end do

! Test to see that first range value is 0.

   if( terx(1) .gt. 0. ) then
      ierror = -18
      return
```

```
   end if

! Determine if terrain profile points are spaced at fixed increments.

   rdif1 = terx(2)-terx(1)
   rfrac = 0.
   rdifsum = rdif1
   do i = 2, itp-1
      rdif2 = amax1( 1.e-3, terx(i+1) - terx(i) )
      rdiff = rdif2 / rdif1
      if( rdiff .gt. rfrac ) rfrac = rdiff
      rdifsum = rdifsum + rdif2
      rdif1 = rdif2
   end do

! If it is determined that terrain points are spaced at fixed range
! increments, then set this increment = RFIX.

   rfix = 0.
   if( rfrac .lt. 1.05 ) rfix = nint( rdifsum / float(itp-1) )

! Test to see if the last range point in the profile meets or exceeds RMAX.
! If not then return error code.

   if( terx(itp) .lt. rmax ) then
      if( lerr6 ) then
         ierror = -6
         return
      end if
   end if

! Determine minimum height of terrain profile.

   hminter = hmax
   do i = 1, itp
      yi = tery(i)
      if( yi .lt. hminter ) hminter = yi
   end do

! Then adjust entire terrain profile by this minimum height HMINTER
! such that this is the new 0 reference.  Get maximum height of terrain,
! store adjusted terrain profile in arrays TX() and TY().


   htermax = 0.
   do i = 1, itp
      tx(i) = terx(i)
      htermax = amax1( tery(i), htermax )
      ty(i) = tery(i) - hminter
   end do

! Add extra point to working terrain profile arrays TX() and TY().

   if( tx(itp) .lt. rmax ) then
      tx(itpa) = rmax * 1.1
   else
      tx(itpa) = tx(itp) * 1.1
   end if
   ty(itpa) = ty(itp)

! Return error code if HMAX does not exceed the maximum height of the
! terrain profile.

   if( htermax .gt. hmax ) then
      ierror = -8
      return
   end if
```

```
   antref = antht + ty(1)

   do i = 1, itpa-1

      y1 = ty(i)
      x1 = tx(i)
      ip1 = i + 1
      y2 = ty(ip1)
      x2 = tx(ip1)

      xdif = x2 - x1
      ydif = y2 - y1
      xdif = amax1( xdif, 1.e-5 )
      slope = ydif / xdif

      slp(i) = slope

! Calculate angle made from each terrain point height to antenna height above
! reference (HMINTER).  Determine maximum propagation angle so that direct ray
! angle will clear highest peak.

      if( y1 .gt. antref ) then
         angle = atan( (y1-antref) / x1 ) !angle from reflected ray
         if( angle .gt. angu ) angu = angle
      end if

   end do

!  Add 1/2 degree to the angle that clears the highest peak.

   angu = angu + hdeg

end if

hmref = hmin - hminter
htlim = hmax - hminter

! Return error if antenna height is greater than maximum plot height.

if( antref .gt. htlim ) ierror = -9

end subroutine terinit
```

A.1.20 Subroutine TROPOINIT

```
!********************** SUBROUTINE TROPOINIT ************************

! Module Name: TROPOINIT

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine initializes all variables and arrays need for
!          troposcatter loss computations.

! Version Number 1.0

! INPUTS:
!   Argument List: NONE
!   Common: AEK2, ANTREF, FREQ, FTER, ITPA, NROUT, NZOUT
!   Data: AEK, EK
!   Public: REFDUM(), RNGOUT(), TX(), TY(), ZOUT()

! OUTPUTS:
!   Argument List: NONE
```

```
!   Common: JT1, JT2, KTR1, THETA1S, TLSTS, R1T, RF, SN1
!   Public: AD1(), ADIF(), D2S(), RDT(), TH1(), THETA0(), THETA2S()

! Modules Included: APM_MOD

! Calling Routines: APMINIT

! Routines Called:
!   APM Specific: ANTPAT
!   Intrinsic: ALOG10, SQRT

! GLOSSARY:

!   Input Variables: See universal glossary for common variables.

!   Output Variables: See universal glossary for common variables.

!   Local Variables:
!       ALD = Log of antenna pattern factor for ALPHAD where ALPHAD here
!             represents lowest direct ray angle in optical region.
!       D1 = Range of each terrain point in meters
!       D1S = Tangent range in meters for source height over smooth
!             surface.
!       D2 = Tangent range in meters for output receiver height Z
!            over smooth surface.
!       FACTR = Antenna pattern factor for angle ALPHAD.
!       H1 = Height of each terrain point in meters
!       QA = Term for determining horizon range.
!       RDHOR1 = Minimum range (in meters) at which diffraction field
!                solutions are applicable and intermediate region ends,
!                for smooth surface and 0 receiver height.
!       SNREF = Surface refractivity.
!       TST = Current largest tangent angle from source.

subroutine tropoinit

use apm_mod

!Initialize surface refractivity.

snref = refdum(0)

!Initialize THETA0 angle.

do i = 1, nrout
   theta0(i) = rngout(i) / aek
end do

!Initialize terms used in calculation of troposcatter loss.

sn1 = .031 - .00232 * snref + 5.67e-6 * snref**2
rf = .0419 * freq
r1t = rf * antref

!Initialize range to tangent point, D1S, and tangent angle,
!THETA1S, for source over smooth surface.

ald = 0.
d1s = sqrt( aek2 * antref )
theta1s = -d1s / aek

!Get antenna pattern loss term, ALD, based on smooth earth tangent
!angle.

alphad = theta1s + 1.e-6
call antpat( alphad, FACTR )
if( factr .ne. 0. ) ald = 20. * alog10( factr )
```

```
!Determine the minimum range, RDHOR1, at which diffraction field
!solutions are applicable and intermediate region ends, for smooth
!surface and 0 receiver height.

qa = 230200. * ( ek**2 / freq )**.3333333
rdhorl = sqrt( aek2 * antref ) + qa     !in meters

!Initialize tangent range and tangent angle, D2S & THETA2S, (for smooth
!surface) for all output receiver heights.

do i = 0, nzout
   z = zout(i)
   if( z .lt. 0. ) cycle
   d2 = sqrt( aek2 * z )
   theta2s(i) = -d2 / aek
   d2s(i) = d2

!Determine minimum range, RDT(), at which diffraction field
!solutions are applicable and intermediate region ends (for smooth
!surface) for all output receiver heights.  Initialize ADIF() for use
!in TROPO routine.

   rdt(i) = rdhorl + d2
   adif(i) = antref - z
end do

!For terrain, determine all increasing tangent ranges and tangent angles,
!AD1() & TH1().

if( fter ) then
   ald = 0.
   j = 0
   tst = theta1s
   do i = 2, itpa
      h1 = ty(i)
      d1 = tx(i)
      ang1 = (h1 - antref) / d1 - d1 / aek2
      if( ang1 .gt. tst ) then
         if(( d1 .gt. d1s ) .and. ( j .eq. 0 )) then
            j = j + 1
            th1(j) = theta1s
            ad1(j) = d1s
         end if
         j = j + 1
         th1(j) = ang1
         ad1(j) = d1
         tst = ang1
      end if
   end do
   if( j .eq. 0 ) then
      j = j + 1
      th1(j) = theta1s
      ad1(j) = d1s
   end if

   ktr1 = j

!Initialize array index counters.

   jt1 = 1
   jt2 = 1
end if

!Initialize troposcatter loss term.

tlsts = 54.9 + 30. * alog10( freq ) - .2 * snref - ald
```

```
end subroutine tropoinit
```

A.1.21 Subroutine XYINIT

```
! ******************************* SUBROUTINE XYINIT ************************

! Module Name: XYINIT

! Module Security Classification: UNCLASSIFIED

! Purpose:  Determines the initial PE starter field.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: ANTHT, DELP, FKO, N, N34, WL, ZMAX
!   Public: FILT()

! OUTPUTS:
!   Argument List: NONE
!   Common: U()

! Modules Used: APM_MOD

! Called Routines: APMINIT

! Routines Called:
!   APM Specific: ANTPAT
!   Intrinsic: ASIN, CMPLX, CONJG, COS, FLOAT, SIN, SQRT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables: NONE

!   Output Variables: NONE

!   Local Variables:
!       ANTKO = Free space wavenumber * antenna height
!       ATTN = Attenuation factor.
!       DTERM = Exponential phase term for real source.
!       DTHETA = Sine of bin angle, i.e., sin(theta), where theta is the
!                transform angle.
!       FACD = Antenna pattern factor for direct angle
!       FACR = Antenna pattern factor for reflected (image) angle
!       PK = Sine of angle.
!       RTERM = Exponential phase term for image source.
!       SGAIN = Normalization factor.
!       ZPK = Phase term for real and image sources.

SUBROUTINE xyinit

use apm_mod

complex rterm, dterm

sgain= sqrt( wl ) / zmax

dtheta = delp / fko
antko = fko * antht

DO I=0,N

   pk = float(i) * dtheta
```

```
   zpk = pk * antko

! Get antenna pattern factors for the direct and reflected rays.

   alphad = asin( pk )
   call antpat( alphad, FACD )
   call antpat( -alphad, FACR )

   rterm = cmplx( cos( zpk ), sin( zpk ) )
   dterm = conjg( rterm )

! Assume perfect conductor, horizontal polarization.

   u(i) = sgain * ( facd * dterm - facr * rterm )

end do

! Filter upper 1/4 of the field.

do i = n34, n
   attn = filt(i-n34)
   u(i) = attn*u(i)
end do

END subroutine xyinit
```

A.2 SUBROUTINE APMSTEP

```
! *********************** SUBROUTINE APMSTEP *******************

! Module Name: APMSTEP

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine advances and computes the propagation loss for
!          one output range step.

! Version Number: 1.0

! INPUTS:
!   Argument List: ISTP
!   Common: GASATT, HTLIM, IHYBRID, IPOL, IZG, KABS, NROUT, NZOUT, RATZ
!   Public: HTFE(), RNGOUT(), RSQRD(), ZOUT()
!   Data: RTST

! OUTPUTS:
!   Argument List: JEND, JSTART, MLOSS(), ROUT
!   Common: JT1, JT2

! Modules Used: APM_MOD

! Calling Routines: MAIN DRIVER PROGRAM

! Routines called:
!   APM Specific: FEM, PESTEP, ROM
!   Intrinsic: AMAX0, NINT

! GLOSSARY: See universal glossary for common variables and parameters.

!   Input Variables:
!       ISTP = Current output range step index.

!   Output Variables:
!       JEND = index at which the valid propagation loss values end.
!       JSTART = index at which the valid propagation loss values begin.
!       ROUT = current output range in meters.
!       MLOSS() = 2-byte integer array containing propagation loss values
!                 in centibels vs. height, at each output range ROUT.
!                 All loss values returned are referenced to height HMIN.

!   Local Variables:
!       JFE = ending index within MLOSS() of FE loss values.
!       JFS = starting index within MLOSS() of FE loss values.
!       JPE = ending index within MLOSS() of PE loss values.
!       JPS = starting index within MLOSS() of PE loss values.
!       JRE = ending index within MLOSS() of RO loss values.
!       JRS = starting index within MLOSS() of RO loss values.
!       LABSCB = Loss due to gaseous absorption in centibels
!       RSQ = Square of output range ROUT

subroutine apmstep( istp, ROUT, MLOSS, JSTART, JEND )

use apm_mod

integer*2 mloss(0:*), labscb

double precision rsq

rout = rngout(istp)
rsq = rsqrd(istp)
```

```
jps = 0
jpe = 0
jrs = 0
jre = 0
jfs = 0
jfe = 0

!Advance and compute the field for one output range step.

call pestep( istp, rout, MLOSS, JPS, JPE )

jstart = jps

if(( ihybrid .eq. 1 ) .and. ( rout .lt. ratz )) then

   if( rout .le. rtst ) then
      jfs = amax0( 0, izg )
      if( ipol .eq. 0 ) jfs = jfs + 1
      jfe = nzout
   else
      if( htfe(istp) .lt. htlim-1.e-3 ) then
         j = nzout
         do while( zout(j) .gt. htfe(istp) )
            j = j - 1
         end do
         jfs = amax0( jpe+1, j+1 )
         jfe = nzout
      end if
   end if

   if( jfe .gt. 0 ) call fem( rout, rsq, MLOSS, JFS, JFE )

! Get loss based on RO calculations from ZOUT(JRS) to ZOUT(JRE).

   if( rout .gt. rtst ) then
      jre = jfs - 1
      if( jre .lt. 0 ) jre = nzout
      if(( jre .eq. 0 ) .and. ( ipol .eq. 0 )) jre = nzout
      jrs = jpe + 1
      if( jpe .eq. 0 ) jrs = 0
      if( jrs .gt. jre ) then
         jrs = 0
         jre = 0
      end if
   end if

   if( jre .gt. 0 ) call rom( istp, rout, MLOSS, jrs, jre )

end if

jend = amax0( jfe, jre, jpe )

!Compute loss due to gaseous absorption and add to propagation loss.

if( kabs .gt. 0 ) then
   do i = jstart, jend
      labscb = nint( rout * gasatt )
      mloss(i) = mloss(i) + labscb
   end do
end if

! Reset counters for calling TROPO from XOSTEP routine.

if( istp .eq. nrout ) then
   jt1 = 1
   jt2 = 1
end if
```

```
end subroutine apmstep
```

A.2.1 Subroutine CALCLOS

```
! **************************** SUBROUTINE CALCLOS ************************

! Module Name: CALCLOS

! Module Security Classification: UNCLASSIFIED

! Purpose: Determines the PE propagation loss at each output range step
!          ROUT and all heights up to ZLIM.

! Version Number: 1.0

! INPUTS:
!   Argument List: ISTP, RLAST
!   Common: DELZ, DR, DZOUT, FTER, HMREF, IHYBRID, IPOL, ITROPO, IXO,
!           NZOUT, RLOG, RLOGLST, RPEST, YCUR, YLAST, ZLIM
!   Public: FSLR(), HLIM(), HTFE(), RNGOUT(), U(), ULST(), ZOUT()

! OUTPUTS:
!   Argument List: JEND, JSTART, MLOSS()
!   Common: IZG
!   Public: FFROUT(), RFAC1(), RFAC2(), RLOSS()

! Modules Used: APM_MOD

! Calling Routines: PESTEP

! Routines called:
!   APM Specific: GETPFAC(function), TROPO
!   Intrinsic: AMAX0, AMAX1, AMIN1, INT, NINT

! GLOSSARY: See universal glossary for common variables, public variables,
!           and parameters.

!   Input Variables:
!       ISTP = index of output range step
!       RLAST = last PE range in meters

!   Output Variables:
!       JEND = index at which valid loss values in MLOSS() ends.
!       JSTART = index at which valid loss values in MLOSS() begin.
!       MLOSS() = 2-byte integer array containing propagation loss values
!                 in centibels vs. height, at each output range ROUT.
!                 All loss values returned are referenced to height HMIN.

!   Local Variables:
!       FF = Propagation factor in dB at range ROUT and specified height.
!       IP1 = Index in array RFAC1() corresponding to ground height at
!             previous PE range.  All array elements in RFAC1() from 1 to
!             IP1 are set equal to PFACMIN.
!       IP2 = Index in array RFAC2() corresponding to ground height at
!             current PE range.  All array elements in RFAC2() from 1 to
!             IP2 are set equal to PFACMIN.
!       PFACMIN = Minimum propagation factor allowed to avoid overflow.
!       RLOSS() = Propagation loss in dB vs. height at range ROUT.
!       ROUT = Current output range in meters.
!       XX = Fractional range at which to interpolate propagation factor
!            for current output range ROUT.
!       YCH = Height of terrain at current PE range relative to
!             reference height HMREF.
!       YLH = Height of terrain at previous PE range relative to
```

```
!              reference height HMREF.
!        ZEND2 = Height at which to stop calculating propagation factor.
!        ZHT = Height at which to compute propagation factor.
!        ZINT = Interpolated ground height at current output range ROUT.

subroutine calclos( rlast, istp, MLOSS, JSTART, JEND )

use apm_mod

integer*2 mloss(0:*)

data pfacmin / 300. /      !Set minimum propagation factor of 300 dB

! Define in-line function for linear interpolation.

plint(pl1, pl2, frac) = pl1 + frac * ( pl2 - pl1 )

rout = rngout(istp)
ych = ycur - hmref
ylh = ylast - hmref

! Get height of ground at output range ROUT and determine number of vertical
! output points that correspond to the ground height.  Fill the loss array
! MLOSS() with zeros to represent ground for those vertical output points.

xx = (rout - rlast) / dr
zint = plint( ylast, ycur, xx )
izg = int( (zint-hmref) / dzout )
do i = 0, izg
   mloss(i) = 0
end do

jstart = amax0( 0, izg )
if( ipol .eq. 0 ) jstart = jstart + 1

! If current output range is greater than RPEST then begin calculation of
! loss values and return them in MLOSS().

if( rout .gt. rpest ) then

! Determine values of array elements corresponding to the ground and set these
! to the minimum propagation factor (-300) for later interpolation.

   ip1 = 0
   ip2 = 0

   if( fter ) then
      ip1 = int( ylh / dzout )
      ip2 = int( ych / dzout )
      ip1 = amax0( 0, ip1 )
      ip2 = amax0( 0, ip2 )

      if( zout(ip1)-ylast .lt. 0. ) ip1 = ip1 + 1
      if( zout(ip2)-ycur .lt. 0. ) ip2 = ip2 + 1

      do i = 0, ip1
         rfac1(i) = pfacmin
      end do
      do i = 0, ip2
         rfac2(i) = pfacmin
      end do
   end if

! Determine height/integer value at which to stop calculating loss.
! NOTE: For terrain cases, ray tracing was performed
! using the direct ray angle and sometimes HLIM(i) may be less than the
! local ground height. The GOTO statement is used just as a safety factor
```

```
! in this case.

   zend1 = amax1( zint, hlim(istp) )
   zend2 = amin1( zlim, zend1 )
   jend = amax0( 0, nint( (zend2-hmref) / dzout ) )

   if( jend .lt. jstart ) goto 5

! Get propagation factor at valid heights from field at previous PE range.

   if( rloglst .gt. 0. ) then
      do i = ip1, jend
         zht = zout(i) - ylast
         rfac1(i) = getpfac( ulst, rloglst, delz, zht )
      end do
   end if

! Get propagation factor at valid heights from field at current PE range.

   do i = ip2, jend
      zht = zout(i) - ycur
      rfac2(i) = getpfac( u, rlog, delz, zht )
   end do

! If using PE model only or PE & XO model, determine what heights in MLOSS()
! will contain invalid loss and set equal to -1.

   if( ihybrid .ne. 1 ) then
      jstart1 = int( (htfe(istp) - hmref) / dzout )
      do i = jstart, jstart1
         mloss(i) = -1
      end do
      jstart = amax0( jstart, jstart1+1)
   end if

! If using full hybrid model or PE & XO model, determine the
! propagation factor at the last point in height in the PE region.  This
! is used for subsequent interpolation in the XO model.

   if( ixo .gt. 0 ) then
      z1 = zlim - ylast
      z2 = zlim - ycur
      rf1 = getpfac( ulst, rloglst, delz, z1 )
      rf2 = getpfac( u, rlog, delz, z2 )
      ff = plint( rf1, rf2, xx )
      ffrout(1,istp) = ff
      ffrout(2,istp) = zlim-zint
  end if

! Interpolate between the current and last PE range to get propagation loss
! at range ROUT.  Compute troposcatter loss for total loss contribution.

   do k = jstart, jend
      if( rloglst .gt. 0. ) then
         ff = plint( rfac1(k), rfac2(k), xx )
         rloss(k) = ff + fslr(istp)
      else
         rloss(k) = rfac2(k) + fslr(istp)
      end if
   end do
   if( itropo .eq. 1 ) call tropo( istp, jstart, jend )
   do k = jstart, jend
      mloss(k) = nint( 10.* rloss(k) )
   end do

 5 continue
```

```
! Fill remainder of array with -1 indicating non-valid loss values.

   jn = jend + 1
   do i = jn, nzout
      mloss(i) = -1
   end do

else

! If current output range is less than RPEST then there are no current valid
! loss values at any height - fill MLOSS() with -1. JSTART and JEND will be
! equal and will have a value of 1 if smooth surface case, otherwise will
! have a value of the nearest integer multiple of DZOUT corresponding to the
! height of the local ground.

   jend = jstart
   do i = jstart, nzout
      mloss(i) = -1
   end do

end if

end subroutine calclos
```

A.2.2 Subroutine DOSHIFT

```
! ************************* SUBROUTINE DOSHIFT *****************************

! Module Name: DOSHIFT

! Module Security Classification: UNCLASSIFIED

! Purpose: Shifts the PE field by the # of bins corresponding to height of
!          the ground.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: DELZ, N, NM1, YCUR, YLAST
!   Public: U()

! OUTPUTS:
!   Argument List: NONE
!   Public: U()

! Modules Used: APM_MOD

! Calling Routines: PESTEP

! Routines called:
!   APM Specific: NONE
!   Intrinsic: ABS, NINT

! GLOSSARY:

!   Input Variables:
!       See universal glossary for common variables

!   Output Variables:
!       See universal glossary for common variables

!   Local Variables:
!       INCR = Integer indicating which direction to shift field U().
!              INCR = 1 -> terrain slope is positive, shift down.
```

```
!              INCR = -1 -> terrain slope is negative, shift up.
!       KBIN = Integer # of bins or mesh heights to shift.
!       YDIF = Height difference between current and last ground elevation.

subroutine doshift

use apm_mod

! Determine # of bins to shift field.

ydif = ycur - ylast
kbin = nint( abs(ydif) / delz )
if( kbin .eq. 0 ) return

! If slope is positive then shift array elements down.

if( ydif .ge. 0. ) then
   incr = 1
   jst = 1
   jend = nm1 - kbin
else

! If slope is negative then shift array elements up.

   incr = -1
   jst = nm1
   jend = kbin + 1
end if

kinc = incr * kbin
do j = jst, jend, incr
   jk = j + kinc
   u(j) = u(jk)
end do

if( incr .gt. 0 ) then
   nst = n - kbin
   do j = nst, nm1
      u(j) = 0.
   end do
else
   do j = 1, kbin
      u(j) = 0.
   end do
end if

end subroutine doshift
```

A.2.3 Subroutine FEM

```
! ************************** SUBROUTINE FEM *************************

! Module Name: FEM

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine determines propagation loss based on flat
!          earth calculations for all output heights specified at each
!          output range ROUT.

! Version Number: 1.0

! INPUTS:
!   Argument List: JFE, JFS, ROUT, RSQ
!   Common: ANTREF, FKO, HTLIM, PLCNST, TWOKA
```

```
!   Public: ZOUTMA(), ZOUTPA()

! OUTPUTS:
!   Argument List: MLOSS()
!   Common: ALPHAD, XREFLECT

! Modules Used: APM_MOD

! Calling Routines: APMSTEP, XOSTEP

! Routines called:
!   APM Specific: ANTPAT, GETREFCOEF
!   Intrinsic: ALOG10, AMAX1, ATAN, COS, DLOG10, NINT, SQRT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ROUT = Current output range in meters.
!       RSQ = Square of output range ROUT

!   Output Variables:
!       JFE = Ending index within MLOSS() of FE loss values.
!       JFS = Starting index within MLOSS() of FE loss values.
!       MLOSS() = 2-byte integer array containing propagation loss values
!                 in centibels vs. height, at each output range ROUT.
!                 All loss values returned are referenced to height HMIN.

!   Local Variables:
!       ALPHAR = Reflected ray angle.
!       DLOSS = Propagation loss in dB.
!       FACD = Antenna pattern factor for direct ray.
!       FACR = Antenna pattern factor for reflected ray.
!       FFAC2 = Square of pattern propagation factor.
!       FFACDB = Pattern propagation factor in dB
!       PHDIF = Total phase difference between direct and reflected ray,
!               including phase change upon reflection.
!       R1 = Path length of direct ray
!       R2 = Path length of reflected ray
!       REFCOEF = Complex reflection coefficient.
!       RMAG = Magnitude of reflection coefficient.
!       RPHASE = Phase of reflection coefficient.
!       RSQK = Current output range squared divided by TWOKA (2*ek*a).
!       ZM = Height of desired output point relative to real source height
!            with earth curvature offset.
!       ZP = Height of desired output point relaive to imaginary source
!            height (for reflected ray) with earth curvature offset.

subroutine fem( rout, rsq, MLOSS, jfs, jfe )

use apm_mod

integer*2 mloss(0:*)

complex refcoef

double precision rsq, rsqk, r1, r2, phdif

rsqk= rsq / twoka
xreflect = 0.

! Begin loop for calculations of FE loss for heights from ZOUT(JFS) to
! ZOUT(JFE).

do i = jfs, jfe

   zm = zoutma(i) - rsqk
   zp = zoutpa(i) - rsqk
```

```
!Determine point of reflection.

   xreflect = rout * antref / zp

! ALPHAD = direct ray angle
! ALPHAR = reflected ray angle (grazing angle = -ALPHAR)

   alphad = atan( zm / rout )
   alphar = atan( zp / rout )

   call antpat( alphad, FACD )
   call antpat( -alphar, FACR )

   r1 = sqrt( zm*zm + rsq )
   r2 = sqrt( zp*zp + rsq )

! Determine reflection coefficient.

   call getrefcoef( alphar, REFCOEF, RMAG, RPHASE )

! Now get total phase lag and compute propagation factor and loss.

   phdif = ( r2 - r1 ) * fko + rphase
   frterm = facr * rmag
   ffac2 = facd*facd + frterm*frterm + 2. * facd * frterm * cos(phdif)

   ffacdb = -10. * alog10( amax1( ffac2, 1.e-25 ) )

   dloss = plcnst + 20. * dlog10( r1 ) + ffacdb
   mloss(i) = nint( 10. * dloss )

end do

end subroutine fem
```

A.2.4 Subroutine FRSTP

```
! ************************** SUBROUTINE FRSTP ******************************

! Module Name: FRSTP

! Module Security Classification: UNCLASSIFIED

! Purpose:  Propagates the field FARRAY() in free space by one range step.
!           If polarization is horizontal, then upon entry FARRAY() is the
!           field array U().  If using vertical polarization, FARRAY() is W().

! Version Number: 1.0

! INPUTS:
!   Argument List: FARRAY()
!   Common: NM1
!   Public: FRSP()

! OUTPUTS:
!   Argument List: FARRAY()
!   Common: NONE

! Calling Routines: PESTEP

! Routines called:
!   APM Specific: FFT
!   Intrinsic: NONE
```

```
! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       FARRAY() = Field array to be propagated one range step in free
!                  space (z-space).

!   Output Variables:
!       FARRAY() = Free-space propagated field (returned in z-space).

!   Local Variables: NONE

subroutine frstp( FARRAY )

use apm_mod

complex farray(0:*)

call fft( FARRAY )            !Transform to Fourier space

DO I = 1, NM1                 !Multiply by free-space propagator
   farray(i) = farray(i) * frsp(i)
end do

call fft( FARRAY )            !Transform back to z-space

end subroutine frstp
```

A.2.5 Subroutine fzlim

```
! ************************* SUBROUTINE FZLIM ************************

! Module Name: FZLIM

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine stores the range, propagation factor in dB,
!          and determines and stores the outgoing propagation angle
!          at the top of the PE region at each range R.

! Version Number 1.0

! INPUTS:
!   Argument List: R, RLAST
!   Common: AATZ, DELZ, DR, FTER, IZ, IZMAX, RATZ, RLOG, RLOGLST, YCUR,
!           YLAST, ZLIM
!   Public: U(), ULST()

! OUTPUTS:
!   Argument List: NONE
!   Common: IZ
!   Public: FFACZ(,)

! Modules Used: APM_MOD

! Calling Routines: PESTEP

! Routines called:
!   APM Specific: GETPFAC(function), SAVEPRO, SPECEST
!   Intrinsic: ABS, AMIN0, AMIN1, SIGN

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       R = Current PE range in meters.
!       RLAST = Previous PE range in meters.
```

```
!   Output Variables: See universal glossary for common variables.

!   Local Variables:
!       ANGDIF = The difference between current outgoing propagation
!                angle and previous angle determined.
!       IZP = Previous index counter in FFACZ().
!       PFDB = Propagation factor in dB at current PE range R at height
!              ZLIM.
!       PFDBLST = Propagation factor in dB at previous PE range RLAST at height
!              ZLIM.
!       PFRATZ = Propagation factor in dB at range RATZ and height ZLIM.
!       THOUT = Outgoing propagation angle determined at top of PE region.

subroutine fzlim( r, rlast )

use apm_mod

! FFACZ(1,I) = propagation factor in dB
! FFACZ(2,I) = range in meters
! FFACZ(3,I) = angle in radians

pfdb = getpfac( u, rlog, delz, zlim-ycur )

if( iz .eq. 1 ) then
   pfdblst = getpfac( ulst, rloglst, delz, zlim-ylast )
   frac = ( ratz - rlast ) / dr
   pfratz = pfdblst + frac * ( pfdb - pfdblst )
   ffacz(1,iz) = pfratz
   ffacz(2,iz) = ratz
   ffacz(3,iz) = aatz
   call savepro
end if

! Perform spectral estimation using top layer from height=JZLIM*DELZ to
! height=(JZLIM-NPNTS)*DELZ.  Determine outgoing propagation angle THOUT.

call specest( THOUT )

iz = iz + 1
iz = amin0( iz, izmax )
ffacz(1,iz) = pfdb
ffacz(2,iz) = r

! Do not let THOUT become greater than the GOOD portion of the maximum
! PE propagation angle.

ffacz(3,iz) = amin1( aatz, thout )

if( iz .ge. 2 ) then

! To avoid extreme "spiking", limit the change in angle values.

   izp = iz - 1
   if( .not. fter ) then
      ffacz(3,iz) = amin1( ffacz(3,izp), thout )
      angdif = ffacz(3,iz) - ffacz(3,izp)
      if( abs(angdif) .gt. 1.e-4 )    &
       ffacz(3,iz) = ffacz(3,izp) + sign(1.,angdif)*1.e-4
   else
      if( iz .le. 10 ) then
         angdif = ffacz(3,iz) - ffacz(3,izp)
          if( abs(angdif) .gt. 1.e-4 )    &
               ffacz(3,iz) = ffacz(3,izp) + sign(1.,angdif)*1.e-4
      end if
   end if
```

```
end if

call savepro

end subroutine fzlim
```

A.2.6 Function GETPFAC

```
! **************************** FUNCTION GETPFAC ***************************

! Module Name: GETPFAC

! Module Security Classification: UNCLASSIFIED

! Purpose:  Performs linear interpolation in height on magnitude of the
!           PE field and then calculates propagation factor in dB.

! Version Number: 1.0

! INPUTS:
!   Argument List: DELZ, HEIGHT, RLOG
!   Common: NONE
!   Public: U()

! OUTPUTS:
!   Function: GETPFAC

! Calling Routines: CALCLOS, FZLIM

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: ALOG10, AMAX1, CABS, FLOAT, INT

! GLOSSARY:

!   Input Variables:
!       DELZ = Bin width in z-space = WL / (2*sin(THETAMAX))
!       RLOG = 10. * alog10( PE range )
!       HEIGHT = receiver height in meters
!       U() = Complex array containing PE field solution.

!   Output Variables:
!       GETPFAC =  Propagation factor at height HEIGHT in dB.

!   Local Variables:
!       FB = Real number of bins corresponding to HEIGHT.
!       FR = Real difference between FB and NB.
!       NB = Integer number of bins corresponding to HEIGHT.
!       NBP1 = NB + 1
!       U0 = Complex field at bin directly below (NB) desired height HEIGHT.
!       U1 = Complex field at bin directly above (NBP1) desired height HEIGHT.
!       PMAG0 = Magnitude of field at bin NB.
!       PMAG1 = Magnitude of field at bin NBP1.
!       PMAG = Interpolated magnitude.
!       PMAGMIN = Lower limit on magnitude of field to avoid underflow/
!                 overflow problems.
!       PFAC = Propagation factor in dB.

function GETPFAC( u, rlog, delz, height )

complex u(0:*), u0, u1

data pmagmin/1.e-10/

fb = height / delz
```

```
nb=int(fb)
fr=fb-float(nb)
nbp1=nb+1

u0=u(nb)
u1=u(nbp1)

pmag0 = cabs( u0 )
pmag1 = cabs( u1 )

pmag = pmag0 + fr * (pmag1 - pmag0)

pmag = amax1( pmag, pmagmin )
pfac = -20.*alog10( pmag ) - rlog
getpfac = amax1( pfac, -200. )

end function getpfac
```

A.2.7 Subroutine GETREFCOEF

```
! *************************** SUBROUTINE GETREFCOEF ***********************

! Module Name: GETREFCOEF

! Module Security Classification: UNCLASSIFIED

! Purpose: Calculates the complex reflection coefficient.

! Version Number: 1.0

! INPUTS:
!   Argument List: ANGLE
!   Common: FREQ, IGR, IPOL, XREFLECT
!   Public: CN2(), RGRND()
!   Parameter: PI

! OUTPUTS:
!   Argument List: REFCOEF, RMAG, RPHASE
!   Common: NONE

! Modules Used: APM_MOD

! Calling Routines: FEM, ROCALC

! Routines called:
!   APM Specific: NONE
!   Intrinsic: ATAN2, CABS, CMPLX, COS, CSQRT, IMAG, REAL, SIN

! GLOSSSARY: See universal glossary for common variables and parameters.

!   Input Variables:
!       ANGLE = grazing angle

!   Output Variables:
!       REFCOEF = complex reflection coefficient
!       RMAG = magnitude of the reflection coefficient
!       RPHASE = phase of the reflection coefficient

!   Local Variables:
!       CRAD = Term used in calculation of reflection coefficient.
!              CRAD = sqrt[ n**2 - (cos(angle))**2 ] where n = index
!              of refraction.
!       RNG2T = Complex dielectric constant applied at the point of
!               reflection.
!       SRAD = Term used in calculation of reflection coefficient.
```

```
!              SRAD = n**2 * sin(angle) where n=index of refraction.

subroutine getrefcoef( angle, REFCOEF, RMAG, RPHASE )

use apm_mod

complex refcoef, crad, srad, rng2t

if( igr .eq. 1 ) then
   rng2t = cn2(1)
else
   k = 1
   do while( rgrnd(k) .lt. xreflect )
      k = k + 1
   end do
   k = amax0( 1, k-1 )
   rng2t = cn2(k)
end if

if( ipol .eq. 1 ) then

! Compute complex reflection coefficient for vertical polarization.

   ctheta = cos( angle )
   stheta = sin( angle )
   crad = csqrt( rng2t - ctheta*ctheta )
   srad = rng2t * stheta
   refcoef = (srad - crad) / (srad + crad)
   rmag = cabs( refcoef )
   rphase = atan2( imag( refcoef ), real( refcoef ) )
else

! Compute complex reflection coefficient for horizontal polarization.

! Calculate finite conductivity ref. coef. for H pol for frequencies
! <= 300 MHz.  Assume perfect conductivity for frequencies > 300 MHz.

   if( freq .le. 300. ) then
      ctheta = cos( angle )
      stheta = sin( angle )
      crad = csqrt( rng2t - ctheta*ctheta )
      refcoef = (stheta - crad) / (stheta + crad)
      rmag = cabs( refcoef )
      rphase = atan2( imag( refcoef ), real( refcoef ) )
   else
      refcoef = cmplx( -1., 0. )
      rmag = 1.
      rphase = pi
   end if
end if

end subroutine getrefcoef
```

A.2.8 Subroutine PESTEP

```
! **************************** SUBROUTINE PESTEP *****************************

! Module Name: PESTEP

! Module Security Classification: UNCLASSIFIED

! Purpose:  Propagates the PE field by one output range step DROUT.

! Version Number: 1.0
```

```
! INPUTS:
!   Argument List: ISTP, ROUT
!   Common: ALPHAV, C1, C2, C1X, C2X, DR, DR2, DZ2, FTER, IG, IGR, IPOL,
!           ITPA, IXO, IZ, IZINC, N, NM1, NPROF, RATZ, RK, RLOG,
!           RMAX, RT, YCUR
!   Public: ENVPR(), RGRND(), ROOT(), ROOTM(), SLP(), TX(), TY(), U()

! OUTPUTS:
!   Argument List: JEND, JSTART, MLOSS
!   Common: C1, C2, IG, RLOG, RLOGLST, YCUR, YCURM, YLAST
!   Public: U(), ULST(), W(), YM()
!   Other:  RLAST, RMID

! Modules Used: APM_MOD

! Calling Routines: APMSTEP

! Routines Called:
!   APM Specific: CALCLOS, DOSHIFT, FRSTP, FZLIM, GETALN, REFINTER,
!                 PHASE2
!   Intrinsic: ALOG10, CMPLX

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ISTP = Index of current output range step.
!       ROUT = Current output range in meters.

!   Output Variables:
!       JEND = Ending index within MLOSS() of PE loss values.
!       JSTART = Starting index within MLOSS() of PE loss values.
!       MLOSS() = 2-byte integer array containing propagation loss values
!                 in centibels vs. height, at each output range ROUT.
!                 All loss values returned are referenced to height HMIN.

!   Local Variables:
!       IZT = Counter used in order to determine when to compute outgoing
!             propagation angle, and save propagation factor and refractivity
!             profile.
!       KT = Counter for terrain profile.
!       R = Current PE range in meters.
!       RLAST = PE range at previous step in meters.
!       RMID = Range at which interpolation for range-dependent refractivity
!              profiles is performed.  This is equal to the range midway
!              between the current and next PE range.
!       SLOPE = Slope of current terrain segment.
! (Note: the following variables are only used for vertical polarization)
!       AR = Complex coefficient of partial linear solution to
!            homogeneous equ.
!       BR = Complex coefficient of partial linear solution to
!            homogeneous equ.
!       ARX = Partial linear solution to homogeneous equ.
!       BRX = Partial linear solution to homogeneous equ.
!       C1C = Summation argument in determining AR.
!       C2C = Summation argument in determining BR.
!       SUM1 = Summation term in determining AR.
!       SUM2 = Summation term in determining BR.
!       UI = U(i).
!       UNMI = U(n-i).

subroutine pestep( istp, rout, MLOSS, JSTART, JEND )

use apm_mod

complex ar, br, sum1, sum2, c1c, c2c, arx, brx

integer*2 mloss(0:*)
```

```
save r, kt, slope, rlast

! Initialize local variables.

if( istp .eq. 1 ) then
   r = 0.
   rlog = 0.
   izt = 0
end if

! Begin loop.

DO while( r .lt. rout )

   if( r .gt. 0. ) ylast = ycur
   rlast = r
   rloglst = rlog

! Store the field arrays of the previous range step for subsequent horizontal
! interpolation at range ROUT.

   do i = 0, n
      ulst(i) = u(i)
   end do

   r = r + dr
   rlog = 10. * alog10( r )
   rmid = r - dr2

   if( fter ) then
      if( rlast .le. 1.e-3 ) then
         slope = slp(1)
         kt = 1
      end if

! Check to see if current range is past a range point in terrain profile.
! If so, increment counter, determine terrain height at current range.

      do while((r .gt. tx(kt+1)) .and. (kt .lt. itpa))
         kt = kt + 1
         slope = slp(kt)
      end do
      ycur = ty(kt) + slope * ( r - tx(kt) )

! Determine height at 1/2 range step - for interpolation on refractivity
! profiles.

      kp = kt
      do while( rmid .lt. tx(kp) )
         kp = kp-1
      end do
      ycurm = ty(kp) + slp(kp) * (rmid - tx(kp))

! Calculate new complex refractive index and impedance term if using vertical
! polarization.

      if(( ipol .eq. 1 ) .and. ( ig+1 .le. igr ))then
         if( r .gt. rgrnd(ig+1) ) then
            ig = ig + 1
            call getaln
         end if
      end if

! Perform boundary shift for terrain case.

      if( slope .lt. 0. ) call doshift
```

```
   end if

   if( ipol .eq. 1 ) then
      do i = 1, nm1
         w(i) = (u(i+1) - u(i-1)) / dz2 + alphav * u(i)
      end do

! Transform W() to p-space, then multiply by free-space propagator,
! then transform back.  Upon return W() is in z-space.

      call frstp( W )

! Propagate C1 and C2 coefficients to new range. NOTE: ONLY FOR SMOOTH
! SURFACE (i.e., no wind speed).

      c1 = c1 * c1x
      c2 = c2 * c2x
   else

! Transform U() to p-space, then multiply by free-space propagator,
! then transform back.  Upon return U() is in z-space.

      call frstp( U )

   end if

! If range-dependent and/or terrain case, then interpolate on profile.

   if(( nprof .gt. 1 ) .or. ( fter )) then
      call refinter( istp, rmid )
      CALL PHASE2
   end if

! This follows steps 9-11 in Kuttler's formulation for vertical
! polarization. (Ref. viewgraphs from 1995 PE Workshop)

   if( ipol .eq. 1 ) then
      ym(0) = cmplx(0.,0.)
      do i = 1, nm1
         ym(i) = dz2 * w(i) + rt * ym(i-1)
      end do

! Compute particular solution.

      u(n) = cmplx(0.,0.)
      do i = 1, N
         nmi = n - i
         u(nmi) = rt * (ym(nmi) - u(nmi+1))
      end do

!At this point U() is the particular solution.
!Determine coefficients AR and BR for homogeneous solution.

      sum1 = .5 * ( u(0) + u(n)*root(n) )
      sum2 = .5 * ( u(0)*rootm(n) + u(n) )
      do i = 1, nm1
         c1c = u(i) * root(i)
         c2c = u(n-i) * rootm(i)

         sum1 = sum1 + c1c
         sum2 = sum2 + c2c
      end do

      ar = c1 - rk * sum1
      br = c2 - rk * sum2

!Now compute total solution as the sum of the particular and
```

```
!homogeneous solutions.

      do i = 0, n
         arx = ar * root(i)
         brx = br * rootm(n-i)
         u(i) = u(i) + arx + brx
      end do
   end if

! Multiply by environment term.

   DO I = 0, nm1
      u(i) = u(i) * envpr(i)
   end do

! Perform boundary shift for terrain case.

   if(( fter ) .and. ( slope .ge. 0. )) call doshift

! Store propagation factor along with current range and outgoing
! propagation angle if using hybrid method (for extended optics).

   if(( ixo .ge. 1 ) .and. ( r .gt. ratz )) then
      if( iz .eq. 1 ) call fzlim( r, rlast )
      izt = izt + 1
      if(( izt .eq. izinc ) .or. ( abs(r-rmax) .lt. dr )) then
         call fzlim( r, rlast )
         izt = 0
      end if
   end if

end do

! Calculate propagation loss at range ROUT.

call calclos( rlast, istp, MLOSS, JSTART, JEND )

end subroutine pestep
```

A.2.9 Subroutine Raytrace

```
!********************** SUBROUTINE RAYTRACE ******************

!      AUTHOR:
!        Herb Hitney
!        Space and Naval Warfare Systems Center, San Diego
!        Tropospheric Branch, Code D883
!      ADDRESS:
!        SPAWARSYSCEN SAN DIEGO D883
!        49170 PROPAGATION PATH
!        SAN DIEGO CA 92152-7385
!        Tel: 619-553-1428  DSN 553-1428
!        Fax: 619-553-1417  DSN 553-1417
!        Email: herb@spawar.navy.mil

! With minor code modifications by:
! Author: Amalia E. Barrios
!         SPAWARSYSCEN SAN DIEGO D883
!         49170 Propagation Path
!         San Diego, CA  92152-7385
!         e-mail: barrios@spawar.navy.mil
!         phone: (619) 553-1429
!         fax: (619) 553-1417

!Module Name: RAYTRACE
```

```
!Module Security Classification: UNCLASSIFIED

! PURPOSE: Computes full raytrace to range ROUT for elevation
!          angle at source height.

!Version Number: 1.0 modified from
!                RPO 1.15B     DATE:  19 August 1996

!INPUTS:
!  Argument list: A, ROUT
!  Common: ISTART, LEVELS
!  Public: GR(), Q(), RM(), ZRT()

!OUTPUT:
!  Argument list: AB, DXDA, ITYPE, PLD, PSI, ZR
!  Common: XREFLECT

!Modules Used: APM_MOD

!CALLING ROUTINES: ROCALC, ROLOSS

!ROUTINES CALLED:
!   APM Specific: NONE
!   Intrinsic: ABS, SIGN, SQRT

!GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       A = elevation angle at source in radians
!       ROUT = terminal range in meters

!   Output Variables:
!       AB = elevation angle at end of ray step in radians
!       DXDA = derivative of ROUT w.r.t. a in meters per radian
!       ITYPE = 0 for direct ray, 1 for reflected ray
!       PLD = optical path length difference from ROUT in meters
!       PSI = 0 for direct ray, grazing angle for ref. ray in radians
!       ZR = terminal height in meters

!   Local Variables:
!       AA = elevation angle at start of ray step in radians
!       DELX = range increment in one ray trace step in meters
!       DELZR = height increment in one ray trace step in meters
!       GOFLAG = logical flag, true normally, false to stop raytrace
!       RAD = radical for square root test in ray trace step
!       XSUM = running sum of range during ray trace in meters
!       XTEMP = temporary range in ray trace step in meters
!       ZLIMIT = maximum height of ray that turns around in meters
!       ZMIN = minimum height of ray that turns around in meters

SUBROUTINE raytrace (rout, a, ZR, AB, DXDA, PLD, PSI, ITYPE)

use apm_mod

LOGICAL goflag

! Set initial conditions at start of ray.

xreflect = 0.
aa = a
i = istart
xsum = 0.
dxda = 0.
pld = 0.
psi = 0.
itype = 0
```

```
goflag = .TRUE.

! Main loop repeats until goflag is false (XSUM = ROUT).

DO WHILE (goflag)

   aa2 = aa**2
   IF (aa .GE. 0.) THEN

         gri = gr(i)
         gri2 = 2. * gri

! Ray is upgoing.

      IF (i .EQ. levels) THEN

! Upgoing ray is in highest layer (last step).

         delx = rout - xsum
         ab = aa + delx * gri
         ab2 = ab**2
         delzr = (ab2 - aa2) / gri2
         zr = zrt(i) + delzr
         dxda = dxda + (a / ab - a / aa) / gri
         pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +        &
              (ab2 * ab - aa2 * aa) / 3.) / gri
         goflag = .FALSE.

      ELSE

! Upgoing ray is not in highest layer.

         rad = aa2 + q(i)
         IF (rad .GE. 0.) THEN

! Upgoing ray penetrates current layer.

            ab = SQRT(rad)
            ab2 = ab**2
            delx = (ab - aa) / gri
            xtemp = xsum + delx
            IF (xtemp .LT. rout) THEN

! Full upgoing step in current layer.

               xsum = xtemp
               dxda = dxda + (a / ab - a / aa) / gri
               pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +   &
                      (ab2 * ab - aa2 * aa) / 3.) / gri
               aa = ab
                     aa2 = aa**2
               i = i + 1
                  gri = gr(i)
                  gri2 = 2. * gri

            ELSE

! Final upgoing step in current layer.

               delx = rout - xsum
               ab = aa + delx * gri
               ab2 = ab**2
               zr = zrt(i) + (ab2 - aa2) / gri2
               dxda = dxda + (a / ab - a / aa) / gri
               pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +   &
                      (ab2 * ab - aa2 * aa) / 3.) / gri
               goflag = .FALSE.
```

```
            END IF

         ELSE

! Upgoing ray turns around in current layer.

            delx = -aa / gri
            xtemp = xsum + delx
            IF (xtemp .LT. rout) THEN

! Full step in upgoing segment.

               xsum = xtemp
               xtemp = xsum + delx
               IF (xtemp .LT. rout) THEN

! Full step in downgoing segment.

                  xsum = xtemp
                  ab = -aa

               ELSE

! Last step in downgoing segment.

                  zlimit = zrt(i) - aa ** 2 / gri2
                  delx = rout - xsum
                  ab = delx * gr(i)
                  delzr = ab ** 2 / gri2
                  zr = zlimit + delzr
                  goflag = .FALSE.

               END IF

            ELSE

! Last step in upgoing segment.

               delx = rout - xsum
               ab = aa + delx * gri
               zr = zrt(i) - (aa ** 2 - ab ** 2) / gri2
               goflag = .FALSE.

            END IF

! Following section applies to all upgoing rays that turn around.

            ab2 = ab**2
            dxda = dxda + (a / ab - a / aa) / gri
            pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +   &
                  (ab2 * ab - aa2 * aa) / 3.) / gri
            aa = ab

         END IF
      END IF
   ELSE

! Ray is downgoing.

      griml = gr(i-1)
      griml2 = 2. * griml
      rad = aa2 - q(i - 1)

      IF (rad .GE. 0.) THEN

! Downgoing ray penetrates current layer.
```

```
        ab = -SQRT(rad)
        delx = (ab - aa) / griml
        xtemp = xsum + delx
        IF (xtemp .LT. rout) THEN

! Full downgoing step in current layer.

           xsum = xtemp
           dxda = dxda + (a / ab - a / aa) / griml
           pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +   &
                (ab ** 3 - aa2 * aa) / 3.) / griml
           aa = ab
                  i = i - 1
           IF (i .EQ. 0) THEN

! Downgoing ray reflects from sea surface.

              itype = 1
              psi = ABS(aa)
              xreflect = xtemp
              xtemp = 2. * xsum
              IF (xtemp .LT. rout) THEN

! Use symmetry concept to double ray path up to source level.

                 aa = -a
                 i = istart
                 xsum = xtemp
                 dxda = 2. * dxda
                 pld = 2. * pld

              ELSE

! Downgoing ray reflects, but symmetry concept is not used.

                 aa = -aa

              END IF
           END IF
          aa2 = aa**2

        ELSE

! Final downgoing step in current layer.

           delx = rout - xsum
           ab = aa + delx * griml
           zr = zrt(i) - (aa2 - ab ** 2) / griml2
           dxda = dxda + (a / ab - a / aa) / griml
           pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +   &
                (ab ** 3 - aa2 * aa) / 3.) / griml
           goflag = .FALSE.

        END IF
     ELSE

! Downgoing ray turns around in current layer.

        delx = -aa / griml
        xtemp = xsum + delx
        IF (xtemp .LT. rout) THEN

! Full step in downgoing segment.

           xsum = xtemp
           xtemp = xsum + delx
```

```
            IF (xtemp .LT. rout) THEN

! Full step in upgoing segment.

               xsum = xtemp
               ab = -aa

            ELSE

! Last step is in upgoing segment.

               zmin = zrt(i) - aa2 / griml2
               delx = rout - xsum
               ab = delx * gr(i - 1)
               delzr = ab ** 2 / griml2
               zr = zmin + delzr
               goflag = .FALSE.

            END IF
         ELSE

! Last step is in downgoing segment.

            delx = rout - xsum
            ab = aa + delx * griml
            delzr = (aa2 - ab ** 2) / griml2
            zr = zrt(i) - delzr
            goflag = .FALSE.

         END IF

! Following section applies to all downgoing rays that turn around.

         dxda = dxda + (a / ab - a / aa) / griml
         pld = pld + ((rm(i) - aa2 / 2.) * (ab - aa) +   &
               (ab ** 3 - aa2 * aa) / 3.) / griml
         aa = ab

      END IF
   END IF
END DO

! Terminal elevation angle ab cannot be zero.

IF (ABS(ab) .LT. 1.E-10) ab = SIGN(1.E-10, ab)

END subroutine raytrace
```

A.2.10 Subroutine REFINTER

```
! ************************* SUBROUTINE REFINTER *****************************

! Module Name: REFINTER

! Module Security Classification: UNCLASSIFIED

! Purpose: Interpolates vertically and horizontally on the refractivity
! profiles.

! Version Number: 1.0

! INPUTS:
!   Argument List: ISTP, RANGE
!   Common: FTER, HMINTER, IS, LVLP, NPROF, RV2, YCURM
!   Public: HMSL(,), REFMSL(,), RNGPROF()
```

```
! OUTPUTS:
!   Argument List: NONE
!   Common: IS, LVLEP, RV2
!   Public: HTDUM(), PROFINT(), REFDUM()

! Modules Used: APM_MOD

! Calling Routines: PESTEP

! Routines Called:
!   APM Specific: REMDUP, PROFREF, INTPROF
!   Intrinsic: NONE

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ISTP = Current output range step index.
!       RANGE = Range for profile interpolation.

!   Output Variables:NONE

!   Local Variables:
!       ICHK = Used to set REFDUM() and HTDUM() to the last input profile
!              specified by REFMSL(,NPROF), HMSL(,NPROF) if RANGE is
!              beyond range of last input profile.
!       RV1 = Range of previous user-input profile.

subroutine refinter( istp, range )

use apm_mod

save j, rv1, ichk
data j, rv1 / 0, 0. /

! One-line interpolation function

pint( p1, p2 ) = p1 + fv * ( p2 - p1 )

if( istp .eq. 1 ) ichk = 0
if(( .not. fter ) .and. ( ichk .eq. 1 )) return
lvlep = lvlp

! If there is a range-dependent refractivity profile then interpolate
! horizontally using the two surrounding profiles at range RANGE with all
! duplicate levels.

if(( nprof .gt. 1 ) .and. ( ichk .eq. 0 )) then
   if( range .gt. rngprof( nprof ) ) then
      ichk = 1
      do i = 0, lvlep
         refdum(i) = refmsl(i,nprof)
         htdum(i) = hmsl(i,nprof)
      end do
   else
      IF( range .gt. rv2 ) then
         j = is
         IS=IS+1
         rv1=rv2
         rv2=rngprof(IS)
      end if

      FV=(range-rv1)/(rv2-rv1)

      do i = 0, lvlep
         refdum(i) = pint( refmsl(i,j), refmsl(i,is) )
         htdum(i) = pint( hmsl(i,j), hmsl(i,is) )
```

```
      end do
   end if

! Now remove all duplicate levels with LVLEP now being the # of points in the
! profile at range RANGE.

   call remdup
   call profref( hminter, 0 )

! At this point REFDUM() and HTDUM(), also HREF() and REFREF(),  are
! referenced to HMINTER.

end if

! Using BS method must determine height and M-unit profiles relative to
! ground, where YCURM is now the height of the local ground above the
! reference height HMINTER.

call profref( ycurm, 1 )

! Interpolate vertically with height.  PROFINT is now an N-point (N=2**NFFT)
! array containing the interpolated M-unit values for the refractivity at
! range RANGE.

call intprof

end subroutine refinter
```

A.2.11 Subroutine REMDUP

```
! ************************* SUBROUTINE REMDUP ****************************

! Module Name: REMDUP

! Module Security Classification: UNCLASSIFIED

! Purpose: Removes duplicate refractivity levels in profile.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: LVLEP
!   Public: HTDUM(), REFDUM()

! OUTPUTS:
!   Argument List: NONE
!   Common: LVLEP
!   Public: HTDUM(), REFDUM()

! Modules Used: APM_MOD

! Calling Routines: REFINIT, REFINTER

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: ABS

! GLOSSARY: See universal glossary for common variables.

!   Input Variables: NONE

!   Output Variables: NONE

subroutine remdup
```

```
use apm_mod

!  Remove all duplicate levels in interpolated profile

i = 0
do while( i .lt. lvlep )
   ht1 = htdum(i)
   ht2 = htdum(i+1)
   if( abs(ht1-ht2) .le. 1.e-3 ) then
      lvlep = lvlep - 1
      do j = i, lvlep
         jp1 = j + 1
         htdum(j) = htdum(jp1)
         refdum(j) = refdum(jp1)
      end do
      i = i - 1
   end if
   i = i + 1
end do

end subroutine remdup
```

A.2.12 Subroutine ROCALC

```
!*********************** SUBROUTINE ROCALC ************************

!      AUTHOR:
!        Herb Hitney
!        Space and Naval Warfare Systems Center San Diego
!        Tropospheric Branch, Code D883
!      ADDRESS:
!        SPAWARSYSCEN SAN DIEGO D883
!        49170 PROPAGATION PATH
!        SAN DIEGO CA 92152-7385
!        Tel: 619-553-1428  DSN 553-1428
!        Fax: 619-553-1417  DSN 553-1417
!        Email: herb@spawar.navy.mil

! With minor code modifications by:
! Author: Amalia E. Barrios
!         SPAWARSYSCEN SAN DIEGO D883
!         49170 Propagation Path
!         San Diego, CA  92152-7385
!         e-mail: barrios@spawar.navy.mil
!         phone: (619) 553-1429
!         fax: (619) 553-1417

! Module Name: ROCALC

! Module Security Classification: UNCLASSIFIED

! PURPOSE: Computes and stores ray-optics components as needed
! to "span" range X.  Arrays use the index K, where the elevation
! angle in radians at the origin is GAMMA = K/1000.  XROP and XRON
! are the ranges less and greater than X, respectively.  IROP and
! IRON are indices of the component arrays that correspond to XROP
! and XRON.  The arrays are: DMAGSQ(,) and RMAGSQ(,) = the magnitude
! squared of the direct and reflected rays; and OMEGA(,) = phase angle
! in rad between direct and reflected rays.  Ray-optics components
! are derived from calls to sub raytrace. Newton's method of
! iteration is used to find the direct and reflected elevation
! angles alphad & alphar. Parallel-ray approximations are used as
! starting values for the highest value of K, otherwise the most
! recent values of ALPHAD & ALPHAR are used to start the iteration.
```

```
! Note that KMINP and KMINN are the minimum K values for good
! solutions at ranges XROP and XRON, and KMAX is the maximum K
! needed to exceed HTLIM at both XROP and XRON.

!Version Number: 1.0 modified from
!                RPO 1.15B     DATE:  19 August 1996

! INPUTS:
!   Argument List: X
!   Common: BW, FKO, HTLIM, HTYDIF, IRON, IROP, KMINN, PSILIM, XRON, XROP,
!           YFREF, ZTOL
!   Parameters: PI

! OUTPUTS:
!   Argument List: NONE
!   Common: DELXRO, DMAGSQ(,), HTYDIF, IRON, IROP, KMAX, KMINN,
!           KMINP, OMEGA(,), RMAGSQ(,), XRON, XROP

! SAVE: DALPHA, FRACRO

! Modules used: APM_MOD

! CALLING ROUTINES: ROM

! ROUTINES CALLED:
!   APM specific: ANTPAT, GETREFCOEF, RAYTRACE
!   Intrinsic: ABS, AMAX1, INT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       X = Current output range in meters

!   Output Variables: NONE

!   Local Variables:
!       ALPHAR = reflected ray source elevation angle in radians
!       BETAD = direct ray terminal elevation angle in radians
!       BETAR = reflected ray terminal elevation angle in radians
!       DALPHA = one half the antenna beamwidth in radians
!       DXDAD = direct ray derivative of range w.r.t. elev angle
!       DXDAR = reflected ray derivative of range w.r.t. elev angle
!       DZDAD = direct ray derivative of height w.r.t. elev angle
!       DZDAR = reflected ray derivative of height w.r.t. elev angle
!       FRACRO = RO range step fraction (0. to .25)
!       FSQD = propagation factor squared for direct ray
!       FSQR = propagation factor squared for reflected ray
!       GMAXDA = term used in computing RO range step fraction
!       ITER = iteration counter (1 to 10)
!       ITYPE = ray type flag (0 = direct, 1 = reflected)
!       PFACD = antenna pattern factor for direct ray
!       PFACR = antenna pattern factor for reflected ray
!       PHI = phase lag of reflection coefficient in radians
!       PLDD = path length difference from x for direct ray
!       PLDR = path length difference from x for reflected ray
!       PSI = grazing angle in radians
!       REFCOEF = complex reflection coefficient
!       RMAG = magnitude of reflection coefficient
!       ZD = terminal height of direct ray in meters
!       ZK = height of kth RO index in meters
!       ZR = terminal height of reflected ray in meters

SUBROUTINE rocalc(x)

use apm_mod

complex refcoef
```

```
SAVE dalpha, fracRO

data deg / .01745 /    !1 degree in radians

! Test if new RO calculations are needed. First time is indicated
! by IROP = -1

DO WHILE (x .GE. xROn)
   IF (iROp .EQ. -1) THEN
      iROp = 1
      iROn = 0
      xROn = x
      kmax = 88
      kminp = 0
      kminn = 0
      fracRO = 0.
      dalpha = bw / 2.
         htydif = htlim - yfref
      IF (dalpha .GT. deg) dalpha = deg
   ELSE
      xROp = xROn
      iROp = 1 - iROp
      iROn = 1 - iROn
      kminp = kminn
      kminn = 0
      kmax = INT(1000. * htydif / xROp) + 2
      IF (kmax .GT. 88) kmax = 88
      IF (fracRO .LT. .25) THEN
         gmaxda = AMAX1((.001 * kmax) / dalpha, 5.0)
         fracRO = 1. / (gmaxda - 1.)
      END IF
      delxRO = fracRO * xROp
      xROn = xROp + delxRO
   END IF

! Set starting conditions corresponding to highest angle.
! Assume parallel direct & reflected rays to start. Note DZDAD
! and DZDAR are the direct and reflected ray derivatives of
! height w.r.t. elevation angle at the source.

   alphad = .001 * kmax
   alphar = -alphad
   CALL raytrace(xROn, alphad, ZD, BETAD, DXDAD, PLDD, PSI, ITYPE)
   dzdad = -betad * dxdad
   CALL raytrace(xROn, alphar, ZR, BETAR, DXDAR, PLDR, PSI, ITYPE)
   dzdar = -betar * dxdar

! Main loop to compute all RO components at height ZK.

   k = kmax
   DO WHILE (k .GE. kminn)
      IF (k .GT. 0) THEN
         zk = xROn * .001 * k

! Loop to find direct ray and components at ZK.

         iter = 0
         DO WHILE (iter .LT. 10)
            iter = iter + 1
            alphad = alphad - (zd - zk) / dzdad
            CALL raytrace (xROn, alphad, ZD, BETAD, DXDAD, PLDD, PSI, ITYPE)
            dzdad = -betad * dxdad

! Test for direct ray not being found.

            IF ((ABS(dzdad) .LT. 1.E-6) .OR. (itype .EQ. 1)) THEN
```

```
              kminn = k + 1
              iter = 10
           END IF

! Test for convergence of direct ray.

           IF (ABS(zk - zd) .LT. ztol) iter = 10
        END DO

! Loop to find reflected ray and components at ZK.

        iter = 0
        DO WHILE (iter .LT. 10)
           iter = iter + 1
           alphar = alphar - (zr - zk) / dzdar
           CALL raytrace(xROn, alphar, ZR, BETAR, DXDAR, PLDR, PSI, ITYPE)
           dzdar = -betar * dxdar

! Test for reflected ray not being found.

           IF ((ABS(dzdar) .LT. 1.E-6) .OR. (itype .EQ. 0)) THEN
              kminn = k + 1
              iter = 10
           END IF

! Test for convergence of reflected ray.

           IF (ABS(zk - zr) .LT. ztol) iter = 10
        END DO

! Test for grazing angle less than limiting value.

        IF (psi .LT. psilim) kminn = k

! Compute magnitude of direct and reflected rays
! based on ray focusing.

        fsqd = ABS(xROn / dzdad)
        fsqr = ABS(xROn / dzdar)

! Adjust magnitude of direct and reflected rays based on
! antenna patterns and reflection coefficient.

        CALL antpat( alphad, PFACD )
        CALL antpat( alphar, PFACR )
        CALL getrefcoef( psi, REFCOEF, RMAG, RPHASE )
        fsqd = fsqd * pfacd ** 2
        fsqr = fsqr * (pfacr * rmag) ** 2

! Store ray-optics components in proper arrays.  Phase lag, OMEGA(,),
! is computed based on total path length difference of the two rays
! plus the reflection coefficient phase lag, PHI.

        dmagsq(iROn, k) = fsqd
        rmagsq(iROn, k) = fsqr
        omega(iROn, k) = (pldr - pldd) * fko + rphase

! Force field to zero at the surface by making magnitudes equal
! and phase lag PI.

     ELSE
        dmagsq(iROn, 0) = fsqd
        rmagsq(iROn, 0) = fsqd
        omega(iROn, 0) = pi
     END IF

! Decrement K index.
```

```
      k = k - 1

   END DO

! End of loop that advances ray optics solution to XRON.

END DO

END subroutine rocalc
```

A.2.13 Subroutine ROLOSS

```
!********************** SUBROUTINE ROLOSS **********************

!      AUTHOR:
!        Herb Hitney
!        Space and Naval Warfare Systems Center San Diego
!        Tropospheric Branch, Code D883
!      ADDRESS:
!        SPAWARSYSCEN SAN DIEGO D883
!        49170 PROPAGATION PATH
!        SAN DIEGO CA 92152-7385
!        Tel: 619-553-1428  DSN 553-1428
!        Fax: 619-553-1417  DSN 553-1417
!        Email: herb@spawar.navy.mil

! With minor executable code modifications by:
! Author: Amalia E. Barrios
!          SPAWARSYSCEN SAN DIEGO D883
!          49170 Propagation Path
!          San Diego, CA  92152-7385
!          e-mail: barrios@spawar.navy.mil
!          phone: (619) 553-1429
!          fax: (619) 553-1417

!Module Name: ROLOSS

!Module Security Classification: UNCLASSIFIED

! PURPOSE: Sets propagation loss in centibels at range ROUT for j from
! JMAX to JMIN based on 3 arrays obtained from sub ROCALC: DMAGSQ(,),
! RMAGSQ(,), and OMEGA(,). The 3 arrays are stored in order of (i,k),
! where i = 0 indicates components at range XROP (<ROUT), and i = 1
! indicates components at range XRON (>ROUT). K is the origin ray
! angle integer index in mrad [i.e. 1000 * angle].  KMINP and KMINN
! are the minimum good values of K at XROP and XRON, and KMAX is the
! maximum value of K where good components are stored at both XROP
! and XRON.

!Version Number: 1.0 modified from
!                RPO 1.15B     DATE:  19 August 1996

! INPUTS :
!   Argument list: ISTP, JMAX, JMIN, ROUT
!   Common: DELXRO, DMAGSQ(,), IRON, IROP, KMAX, KMINN,
!           KMINP, OMEGA(,), RMAGSQ(,), XROP
!   Public: FSLR(), ZRO()

! OUTPUTS:
!   Argument list: LOSSCB
!   Common: NONE

! SAVE: DANGHI, DANGLO, DFSDHI, DFSDLO, DFSRHI, DFSRLO
```

```
!Modules Used: APM_MOD

! CALLING ROUTINES: ROM

! ROUTINES CALLED:
!   APM Specific: NONE
!   Intrinsic: ABS, ALOG10, AMAX1, COS, INT, NINT, SQRT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ISTP = Current output range step index.
!       JMAX = Ending index within LOSSCB() of RO loss values.
!       JMIN = Starting index within LOSSCB() of RO loss values.
!       ROUT = Current output range in meters.

!   Output Variables:
!       LOSSCB() = Array containing propagation loss values in
!                  centibels vs. height, at each output range ROUT.
!                  I.e., LOSSCB(J) = propagation loss * 10 at output
!                  height ZOUT = J*DZOUT. All loss values returned
!                  are referenced to height HMIN.

!   Local Variables:
!       ANG = phase angle for computing FSQ in radians
!       ANGHI = phase angle above desired point in radians
!       ANGLO = phase angle below desired point in radians
!       DANGHI = diff. in phase angle along RO step above desired point
!       DANGLO = diff. in phase angle along RO step below desired point
!       DFSDHI = diff. in dir. mag**2 along RO step above desired point
!       DFSDLO = diff. in dir. mag**2 along RO step below desired point
!       DFSRHI = diff. in ref. mag**2 along RO step above desired point
!       DFSRLO = diff. in ref. mag**2 along RO step below desired point
!       FK = floating value of K index at jth output point
!       FFAC =  propagation factor in dB
!       FSDHI = direct ray magnitude squared above desired point
!       FSDLO = direct ray magnitude squared below desired point
!       FSQ = propagation factor squared at desired point
!       FSQD = direct ray magnitude squared at desired point
!       FSQR = reflected ray magnitude squared at desired point
!       FSRHI = reflected ray magnitude squared above desired point
!       FSRLO = reflected ray magnitude squared below desired point
!       KHI = K index above desired point
!       KLO = K index below desired point
!       KLOTMP = temporary KLO value
!       RATIOK = fraction of one K index (0. to 1.)
!       RATIOX = fraction of current RO range step (0. to 1.)

SUBROUTINE roloss ( istp, rout, jmin, jmax, LOSSCB )

use apm_mod

INTEGER*2 losscb(0:*)

SAVE danghi, danglo, dfsdhi, dfsdlo, dfsrhi, dfsrlo

! Compute free-space loss term and ratio of distance from last RO
! range to RO range increment. Set starting value of KLO to KMAX.

ratiox = (rout - xROp) / delxRO
klo = kmax
fkro = 1000. / rout

! Loop to compute loss for all J from JMAX to JMIN.

DO j = jmax, jmin, -1
```

```
! Compute floating (non-integer) value of K corresponding to J, and
! integer value of K just below floating K. Test to see if this value
! is less than the previous value of KLO.

   fk = fkro * zro(j)
   klotmp = INT(fk)

   IF (klotmp .LT. klo) THEN

! Set new KLO and KHI.

      klo = klotmp
      khi = klo + 1

! If KLO is greater than or equal to the minimum K at range XROP
! and XRON, then compute new differences in components between
! XRON and XROP at index KLO. Otherwise old values will be used.

      IF ((klo .GE. kminp) .AND. (klo .GE. kminn)) THEN
         dfsdlo = dmagsq(iROn, klo) - dmagsq(iROp, klo)
         dfsrlo = rmagsq(iROn, klo) - rmagsq(iROp, klo)
         danglo = omega(iROn, klo) - omega(iROp, klo)
      END IF

! If KHI is greater than or equal to the minimum K at range XROP
! and XRON, then compute new differences in components between
! XRON and XROP at index KHI. Otherwise old values will be used.

      IF ((khi .GE. kminp) .AND. (khi .GE. kminn)) THEN
         dfsdhi = dmagsq(iROn, khi) - dmagsq(iROp, khi)
         dfsrhi = rmagsq(iROn, khi) - rmagsq(iROp, khi)
         danghi = omega(iROn, khi) - omega(iROp, khi)
      END IF

! If KLO is greater than or equal to the minimum K at XROP, then
! compute new components at range ROUT at index KLO by linear inter-
! polation from range XROP at KLO.  Otherwise interpolate backwards
! from XRON at KLO.

      IF (klo .GE. kminp) THEN
         fsdlo = dmagsq(iROp, klo) + ratiox * dfsdlo
         fsrlo = rmagsq(iROp, klo) + ratiox * dfsrlo
         anglo = omega (iROp, klo) + ratiox * danglo
      ELSE
         ratioxm1 = 1. - ratiox
         fsdlo = dmagsq(iROn, klo) + ratioxm1 * dfsdlo
         fsrlo = rmagsq(iROn, klo) + ratioxm1 * dfsrlo
         anglo = omega (iROn, klo) + ratioxm1 * danglo
      END IF

! If KHI is greater than or equal to the minimum K at XROP, then
! compute new components at range ROUT at index KHI by linear inter-
! polation from range XROP at KHI.  Otherwise interpolate backwards
! from XRON at KHI.

      IF (khi .GE. kminp) THEN
         fsdhi = dmagsq(iROp, khi) + ratiox * dfsdhi
         fsrhi = rmagsq(iROp, khi) + ratiox * dfsrhi
         anghi = omega (iROp, khi) + ratiox * danghi
      ELSE
         ratioxm1 = 1. - ratiox
         fsdhi = dmagsq(iROn, khi) + ratioxm1 * dfsdhi
         fsrhi = rmagsq(iROn, khi) + ratioxm1 * dfsrhi
         anghi = omega (iROn, khi) + ratioxm1 * danghi
      END IF

   END IF
```

```
   ratiok = fk - klo
   fsqd = fsdlo + ratiok * (fsdhi - fsdlo)
   fsqr = fsrlo + ratiok * (fsrhi - fsrlo)
   ang  = anglo + ratiok * (anghi - anglo)

! Compute square of propagation factor.

   fsq = ABS(fsqd + fsqr + 2. * SQRT(ABS(fsqd * fsqr)) * COS(ang))

! Convert FSQ to propagation factor in dB.  Limit to -200 dB.

   ffac = 10. * alog10(amax1(1.e-25, fsq))

! Compute and store propagation loss in terms of closest
! integer centibel (cB).

   dloss = fslr(istp) - ffac
   losscb(j) = nint( 10. * dloss )

END DO

END subroutine roloss
```

A.2.14 Subroutine ROM

```
!************************* SUBROUTINE ROM **************************

!Module Name: ROM

!Module Security Classification: UNCLASSIFIED

! Purpose: This routine serves as a one-call routine for the ray optics
!          model.  It performs the ray optics calculations by calls
!          to ROCALC and determines the loss at specified height output
!          points by calls to ROLOSS.

!Version Number: 1.0

!INPUTS:
!   Argument List: ISTP, ROUT, JRS, JRE
!   Common: NONE

!OUTPUTS:
!   Argument List: MLOSS
!   Common: NONE

!Modules Used: NONE

!Calling Routines:
!   APM Specific: APMSTEP, XOSTEP
!   Intrinsic: NONE

!Routines called: ROCALC, ROLOSS

!GLOSSARY: See universal glossary for parameters.

!   Input Variables:
!       ISTP = Current output range step index.
!       JRS = Starting index within MLOSS() of RO loss values.
!       JRE = Ending index within MLOSS() of RO loss values.
!       ROUT = Current output range in meters.

!   Output Variables:
!       MLOSS() = Array containing propagation loss values in
```

```
!                  centibels vs. height, at each output range ROUT.
!                  I.e., MLOSS(J) = propagation loss * 10 at output
!                  height ZOUT = J*DZOUT. All loss values returned
!                  are referenced to height HMIN.

!   Local Variables: NONE

subroutine rom( istp, rout, MLOSS, jrs, jre )

integer*2 mloss(0:*)

call rocalc( rout )
call roloss( istp, rout, jrs, jre, MLOSS )

end subroutine rom
```

A.2.15 Subroutine SAVEPRO

```
! ******************************* SUBROUTINE SAVEPRO ************************

! Module Name: SAVEPRO

! Module Security Classification: UNCLASSIFIED

! Purpose: Saves the refractivity profiles at each range step from the top of
!          the PE region to the maximum user-specified height.  For use only
!          when using the hybrid model.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: IZ, LVLEP, ZLIM
!   Public: HTDUM(), REFDUM()

! OUTPUTS:
!   Argument List: NONE
!   Common: NONE
!   Public: GRAD(,), HTR(,), LVL()

! Modules Used: APM_MOD

! Calling Routines: FZLIM

! Routines Called:
!   APM Specific: NONE
!   Intrinsic: ABS, SIGN

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       R = Current PE range in meters

!   Output Variables: NONE

!   Local Variables:
!       G = Gradient of current refractivity profile level.
!       NEWL = Number of levels in refractivity profile from top of PE
!              region to maximum height.

subroutine savepro

use apm_mod

! Determine at what index of current profile to begin storing from height
```

```
! ZLIM.

i = 0
do while( zlim .gt. htdum(i) )
   i = i + 1
end do
i = i - 1
newl = -1

! Store gradients and height levels from this index level - I to LVLEP-1.

do j = i, lvlep-1
   jp1 = j + 1
   rm1 = refdum(j)
   rm2 = refdum(jp1)
   h1 = htdum(j)
   h2 = htdum(jp1)
   g = ( rm2 - rm1 ) / ( h2 - h1 )
   if( abs( g ) .lt. 1.e-3 ) g = sign( 1., g )*1.e-3
   newl = newl + 1
   grad(newl,iz) = g * 1.e-6      ! for ray trace formulas
   htr(newl,iz) = h1
end do
newl = newl + 1
htr(newl,iz) = htdum(lvlep)
lvl(iz) = newl

end subroutine savepro
```

A.2.16 Subroutine SPECEST

```
!*************************** SUBROUTINE SPECEST ************************

! Module Name: SPECEST

! Module Security Classification: UNCLASSIFIED

! Purpose: Determines the outward propagation angle THOUT based on spectral
!          estimation of the topmost layer of the field still
!          within the "good" part of the transform.  Looks at the field
!          from height=JZLIM*DELZ to height=(JZLIM-NPNTS)*DELZ.

! Version Number: 1.0

! INPUTS:
!   Argument List: NONE
!   Common: DELZ, JZLIM, LNP, NP34, NPNTS, NS, XOCON, YCUR
!   Public: FILTP(), U()

! OUTPUTS:
!   Argument List: THOUT
!   Common: NONE
!   Public: SPECTR(), XP(), YP()

! Modules Used: APM_MOD

! Calling Routine: FZLIM

! Routines Called:
!   APM Specific: SINFFT(in module APM_MOD)
!   Intrinsic: ALOG10, AMAX1, ASING, FLOAT, IMAG, NINT, REAL, SQRT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables: NONE
```

```
!   Output Variables:
!       THOUT = outward propagation angle in radians at top of PE height
!               region.

!   Local Variables:
!       AMP = Field magnitude with lower limit of 1.e-10.
!       ATTN = Filter factor - used for filtering field before transforming.
!       IPEAK = Bin # in SPECTR() corresponding to the peak magnitude.
!       K = Bin # at which to start storing PE field.  Points from K to
!           K-NPNTS are stored in XP() and YP().
!       PAVG = 3-pt average magnitude.
!       PEAK = Peak magnitude.
!       PP = Field magnitude.

subroutine specest( THOUT )

use apm_mod

! Store upper NPNTS of U() in XP() and YP().

k = jzlim - nint( ycur / delz )
do i = 0, npnts-1
   xp(i) = real(u(k))
   yp(i) = imag(u(k))
   k = k - 1
end do
do i = np34, npnts
   attn = filtp(i-np34)
   xp(i)=attn*xp(i)
   yp(i)=attn*yp(i)
end do

! Zero pad.

do i = npnts+1, ns-1
   xp(i) = 0.
   yp(i) = 0.
end do

! Transform to obtain spectral field

call sinfft( lnp, xp )
call sinfft( lnp, yp )

! Determine amplitude

do i = 0, ns-1
   xpi = xp(i)
   ypi = yp(i)
   pp = sqrt( xpi*xpi + ypi*ypi )
   amp = amax1(1.e-10, pp)
   spectr(i) = 10.* alog10(amp)
end do

! Perform a 3-point average and look for amplitude peak.

ipeak = 0
peak = -200.
do i = 2, ns-1
   p1 = spectr(i-1)
   p = spectr(i)
   p2= spectr(i+1)
   pavg = (p1 + p + p2) / 3.
   if( pavg .gt. peak) then
      ipeak = i
      peak = pavg
```

```
   end if
end do

! Determine angle from bin# IPEAK where peak occurs.

thout = asin( xocon * float(ipeak) )

end subroutine specest
```

A.2.17 Subroutine TROPO

```
!************************** SUBROUTINE TROPO ***************************

! Module Name: TROPO

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine determines the loss due to troposcatter and
!          computes the appropriate loss from troposcatter loss and the
!          propagation loss beyond the radio horizon.

! Version Number: 1.0

! INPUTS:
!   Argument List: ISTP, JS, JE
!   Common: AEK2, FTER, ITPA, JT1, JT2, KTR1, R1T, RF, THETA1S,
!           TLSTS
!   Public: AD1(), ADIF(), D2S(), RDT(), RLOGO(), RLOSS(), RNGOUT(), TH1(),
!           THETA0(), THETA2S(), TX(), TY(), ZOUT()
!   Data: EK

! OUTPUTS:
!   Argument List: NONE
!   Common: NONE
!   Public: RLOSS()

! Modules Used: APM_MOD

! Calling Routines: CALCLOS, EXTO

! Routines Called:
!   APM Specific:
!   Intrinsic: ALOG10, AMAX0, AMAX1, AMIN1, EXP

! GLOSSARY:

!   Input Variables:
!       ISTP = Current output range step index.
!       JS = Starting index in ZOUT() for troposcatter calculations.
!       JE = Ending index in ZOUT() for troposcatter calculations.
!       RLOSS() = Propagation loss in dB vs. height at range ROUT.

!   Output Variables
!       RLOSS() = Propagation/troposcatter loss in dB vs. height at range ROUT.

!   Local Variables:
!       AL = Angle defined by equ. 115 in EREPS 3.0 User's Manual
!            NRaD TD 2648, pp. 105.
!       ALD = Log of antenna pattern factor for ALPHAD where ALPHAD here
!             represents lowest direct ray angle in optical region.
!       BE = Angle defined in equ. 116 in EREPS 3.0 User's Manual
!            NRaD TD 2648, pp. 105.
!       BIGH = Frequency gain function defined in equ. 119 in EREPS 3.0
!              User's Manual NRaD TD 2648, pp. 106.
!       CT1 = Quantity defined in equ. 124 in EREPS 3.0 User's Manual
```

```
!            NRaD TD 2648, pp. 106.
!       CT2 = Quantity defined in equ. 125 in EREPS 3.0 User's Manual
!             NRaD TD 2648, pp. 106.
!       D1 = Range from source to tangent point in meters.
!       D2 = Range from receiver to tangent point in meters.
!       DELHO = Frequency gain function correction term defined in equ.
!               127 in EREPS 3.0 User's Manual NRaD TD 2648, pp. 106.
!       ETAS = Quantity defined in equ. 126 in EREPS 3.0 User's Manual
!              NRaD TD 2648, pp. 106.
!       H0 = Effective scattering height - defined in equ. 109 in
!            EREPS 3.0 User's Manual NRaD TD 2648, pp. 105.
!       HOR1 = Quantity defined in equ. 120 in EREPS 3.0 User's Manual
!              NRaD TD 2648, pp. 106.
!       HOR2 = Quantity defined in equ. 121 in EREPS 3.0 User's Manual
!              NRaD TD 2648, pp. 106.
!       JZ = Current output height index.
!       QT = Quantity defined in equ. 128 in EREPS 3.0 User's Manual
!            NRaD TD 2648, pp. 107.
!       R1 = Quantity defined in equ. 122 in EREPS 3.0 User's Manual
!            NRaD TD 2648, pp. 106.
!       R2 = Quantity defined in equ. 123 in EREPS 3.0 User's Manual
!            NRaD TD 2648, pp. 106.
!       ROUT = Current output range in meters.
!       ROUT3 = Current output range in km.
!       S = Quantity defined equ. 110 in EREPS 3.0 User's Manual
!           NRaD TD 2648, pp. 105.
!       THETA = Common volume scattering angle in radians.
!       THETA1 = Tangent angle from source height.
!       THETA2 = Tangent angle from receiver height.
!       TLOSS = Troposcatter loss in dB.
!       TLST = Troposcatter loss term.

subroutine tropo( istp, js, je )

use apm_mod

rout = rngout(istp)

!For smooth surface, initialize tangent angle for source height and
!initialize troposcatter loss term, plus other variables dependent only
!on range.

theta1 = theta1s
tlst = tlsts
theta02 = theta0(istp) * .5
rout3 = rout * 1.e-3

!If terrain case, determine tangent angle from source and initialize
!counter for receiver case.

if( fter ) then
   if( rout .lt. ad1(1) ) then
      return
   else
      do while(( rout .gt. ad1(jt1) ) .and. ( jt1 .le. ktr1 ))
         jt1 = jt1 + 1
      end do
      jt1 = amax0( 1, jt1-1 )
      d1 = ad1(jt1)
      theta1 = th1(jt1)
   end if
   do while(( rout .gt. tx(jt2) ) .and. ( jt2 .le. itpa ))
      jt2 = jt2 + 1
   end do
   j2m = jt2 - 1
end if
```

```
do jz = js, je

!For smooth surface, if current output range is less than minimum
!diffraction field range, then still in interference region - exit.

   if(( rout .lt. rdt(jz) ) .and. ( .not. fter )) return

!For smooth surface, initialize tangent angle for receiver height.

   theta2 = theta2s(jz)

   if( fter ) then

!If terrain case, determine tangent angle from receiver.

      d2 = d2s(jz)
      do i = j2m, 1, -1
         h2 = ty(i)
         rx = tx(i)
         r2 = rout - rx
         ang2 = (h2 - zout(jz)) / r2 - r2 / aek2
         if( ang2 .gt. theta2 ) then
            theta2 = ang2
            d2 = r2
         end if
      end do
      if( rout .lt. d1+d2+1.e-2 ) return

!Get antenna pattern loss term, ALD, based on tangent angle from
!source over terrain.

      alphad = theta1 + 1.e-6
      call antpat( alphad, FACTR )
      if( factr .ne. 0. ) ald = 20. * alog10( factr )

!Adjust troposcatter loss term.

      tlst = tlsts - ald
   end if

!Determine common volume scattering angle.

   theta = theta0(istp) + theta1 + theta2

   antdifr = adif(jz) / rout

!Determine angles illustrated and defined in equs. 115 and 116 in
!EREPS 3.0 User's manual.

   al = theta02 + theta1 + antdifr
   be = theta02 + theta2 - antdifr

   s = amin1( amax1( .1, al / be ), 10. )

!Get effective scattering height, H0.

   h0 = s * rout3 * theta / ( 1. + s )**2

!The following variables are determined to compute the frequency gain
!function BIGH.  All variables are defined in equs. 119-128 in EREPS
!3.0 user's manual.

   etas = .5696 * h0 * ( 1. + sn1 * exp(-3.8e-6 * h0**6) )
   etas = amin1( amax1( .01, etas ), 5. )

   ct1 = 16.3 + 13.3*etas
   ct2 = .4 + .16*etas
```

```
   r1 = amax1( .1, r1t * theta )
   r2 = amax1( .1, rf * zout(jz) * theta )
   hor1 = amax1( 0., ct1 * (r1 + ct2)**(-ek) )
   hor2 = amax1( 0., ct1 * (r2 + ct2)**(-ek) )

   qt = amin1( amax1( .1, r2 / s / r1 ), 10. )

   delho = 6.*(.6 - alog10(etas)) * alog10(s) * alog10(qt)

   hp = (hor1 + hor2) / 2.
   delho = amin1( hp, delho )
   if( delho+hp .lt. 0. ) delho = -hp
   bigh = hp + delho

!Troposcatter loss is computed.

   tloss = tlst + 573. * theta + rlogo(istp) + bigh

!Troposcatter loss is compared to propagation loss.  If the difference
!between the propagation loss and troposcatter loss is less than 18 dB,
!then a method of bold interpolation is used to smoothly combine the 2
!losses.  If the difference is greater than 18 dB then lesser of the 2
!losses is used.

   dif = rloss(jz) - tloss
   if( dif .ge. 18. ) then
      rloss(jz) = tloss
   elseif( dif .ge. -18. ) then
      rloss(jz) = rloss(jz) - 10.*alog10( 1. + 10.**(.1*dif) )
   end if
end do

end subroutine tropo
```

A.3 SUBROUTINE XOINIT

```
! ************************ SUBROUTINE XOINIT ************************

! Module Name: XOINIT

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine initializes the range, height and angle arrays
!          in preparation for XOSTEP.  It performs 2 passes on a 10-pt
!          smoothing average to smooth the propagation angles.

! Version Number: 1.0

! INPUTS:
!   Argument List: IXOSTP, JEND
!   Common: FTER, IZ, ZLIM
!   Public: FFACZ(,)

! OUTPUTS:
!   Argument List: JXSTART, IERROR
!   Common: NONE
!   Public: CURANG(), CURHT(), CURNG(), HTOUT() ,IGRD(), PRFAC()

! Modules Included: APM_MOD

! Calling Routines: MAIN DRIVER PROGRAM

! Routines Called:
!   APM Specific: SMOOTH
!   Intrinsic: NONE

! GLOSSARY: See universal glossary for common variables and parameters.

!   Input Variables:
!       JEND = Output index in MLOSS() where loss values calculated from
!              PE model ends.

!   Output Variables:
!       JXSTART = Output index in MLOSS() where loss values calculated from
!                 from XO model begins.

!   Local Variables:
!       AX = Running sum of first 3 angles computed. Used only for smooth
!            surface case.
!       DUM = Dummy array for CURANG().

subroutine xoinit( ixostp, jend, JXSTART, IERROR )

use apm_mod

real, allocatable :: dum(:)

if( ixostp .gt. 0 ) then

   if( allocated( curang ) ) deallocate( curang, stat=ierror )
   allocate( curang(iz), stat=ierror )
   if( ierror .ne. 0 ) return
   curang = ffacz(3,:)

   if( allocated( curht ) ) deallocate( curht, stat=ierror )
   allocate( curht(iz), stat=ierror )
   if( ierror .ne. 0 ) return

! Initialize so that ray tracing in subroutine EXTO begins at height
```

```
! ZLIM with the first gradient at index 0 in array GRAD(,).

   curht = zlim

   if( allocated( curng ) ) deallocate( curng, stat=ierror )
   allocate( curng(iz), stat=ierror )
   if( ierror .ne. 0 ) return
   curng = ffacz(2,:)

   if( allocated( igrd ) ) deallocate( igrd, stat=ierror )
   allocate( igrd(iz), stat=ierror )
   if( ierror .ne. 0 ) return
   igrd = 0.

   if( allocated( htout ) ) deallocate( htout, stat=ierror )
   allocate( htout(iz), stat=ierror )
   if( ierror .ne. 0 ) return
   htout = 0.

   if( allocated( prfac ) ) deallocate( prfac, stat=ierror )
   allocate( prfac(iz), stat=ierror )
   if( ierror .ne. 0 ) return
   prfac = 0.

   if( allocated( dum ) ) deallocate( dum, stat=ierror )
   allocate( dum(iz), stat=ierror )
   if( ierror .ne. 0 ) return
   dum = 0.

   if( fter ) then

! Now perform 1st smoothing on entire angle array.

      call smooth( curang, iz, 10, DUM )

! Now perform 2nd smoothing on entire angle array.

      call smooth( dum, iz, 10, CURANG )

   end if

   jxstart = jend + 1

   deallocate( dum )

end if

end subroutine xoinit
```

A.3.1 Subroutine SMOOTH

```
! ************************* SUBROUTINE SMOOTH **********************

! Module Name: SMOOTH

! Module Security Classification: UNCLASSIFIED

! Purpose: Performs IAV-pt average smoothing.

! Version Number: 1.0

! INPUTS:
!   Argument List: ARBEF(), IAV, IZ
!   Common: NONE
```

```
! OUTPUTS:
!   Argument List: ARAFT()
!   Common: NONE

! Modules Used: NONE

! Calling Routines: XOINIT

! Routines called:
!   APM Specific: NONE
!   Intrinsic:  AMAX0, AMIN0, FLOAT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ARBEF = array before smoothing.
!       IAV = # of points in which to take average smoothing.
!       IZ = # of points in array.

!   Output Variables:
!       ARAFT = array after smoothing.

!   Local Variables:
!       AX = Temporary averaged value.
!       IA = Number of points past and previous to the desired point to
!            include in averaging.
!       NAX = Current number of points averaged.

subroutine smooth( arbef, iz, iav, ARAFT )

dimension arbef(*), araft(*)

do k = 1, iz
   ia = amin0( k, iav, iz-k )
   j = amax0( 1, k-ia)
   l = amin0( iz, k+ia )
   ax = 0.
   nax = l - j + 1
   do i = j, l
      ax = ax + arbef(i)
   end do
   ax = ax / float(nax)
   araft(k) = ax
end do

end subroutine smooth
```

A.4 SUBROUTINE XOSTEP

```
!************************* SUBROUTINE XOSTEP ****************************

! Module Name: XOSTEP

! Module Security Classification: UNCLASSIFIED

! Purpose: Calculates loss values in the height region above
!          the maximum height of the PE model for one range step.

! Version Number: 1.0

! INPUTS:
!   Argument List: ISTP, JXSTART
!   Common: GASATT, HTLIM, IHYBRID, KABS, NZOUT
!   Public: HTFE(), RNGOUT(), RSQRD(), ZOUT()

! OUTPUTS:
!   Argument List: JXEND, MLOSS(), ROUT
!   Common: NONE

! Modules Used: APM_MOD

! Calling Routines: MAIN DRIVER PROGRAM

! Routines Called:
!   APM Specific: EXTO, FEM, ROM
!   Intrinsic: AMIN0, NINT

! GLOSSARY: See universal glossary for common variables.

!   Input Variables:
!       ISTP = Current output range step index.
!       JXSTART = Output index in MLOSS() where loss values calculated
!                 from FE/RO/XO model begins.

!   Output Variables:
!       JXEND = Index at which the valid propagation loss values end.
!       ROUT = Current output range in meters.
!       MLOSS() = 2-byte integer array containing propagation loss values
!                 in centibels vs. height, at each output range ROUT.
!                 All loss values returned are referenced to height HMIN.

!   Local Variables:
!       JFE = ending index within MLOSS() of FE loss values.
!       JFS = starting index within MLOSS() of FE loss values.
!       JRE = ending index within MLOSS() of RO loss values.
!       JRS = starting index within MLOSS() of RO loss values.
!       LABSCB = Loss due to gaseous absorption in centibels
!       RSQ = Square of output range ROUT

subroutine xostep( istp, ROUT, MLOSS, jxstart, JXEND )

use apm_mod

integer*2 mloss(0:*), labscb

double precision rsq

if( ihybrid .eq. 0 ) return

jfs = 0
jfe = 0
jrs = 0
```

```
jre = 0

rout = rngout(istp)
rsq = rsqrd(istp)

do j = jxstart, nzout
   mloss(j) = -1
end do

! Perform extended optics calculations.
! JXE = ending index within MLOSS() of XO loss values.

call exto( istp, rout, MLOSS, jxstart, JXE )

if( ihybrid .eq. 1 ) then
   if( htfe(istp) .lt. htlim-1.e-3 ) then
      j = nzout
      do while( zout(j) .gt. htfe(istp) )
         j = j - 1
      end do
      jfs = amax0( jxe+1, j+1 )
      jfe = nzout
   end if

   if( jfe .gt. 0 ) call fem( rout, rsq, MLOSS, JFS, JFE )

! Perform RO calculations if necessary
! JRS = starting index within MLOSS() of RO loss values.
! JRE = ending index within MLOSS() of RO loss values.

   if( jxe .lt. nzout ) then
      jre = jfs - 1
      if( jre .lt. 0 ) jre = nzout
      jrs = jxe + 1
      if( jrs .gt. jre ) then
         jrs = 0
         jre = 0
      end if
      if( jre .gt. 0 ) call rom( istp, rout, MLOSS, jrs, jre )
   end if
end if

jxend = amax0( jxe, jfe, jre )

if( kabs .gt. 0 ) then
   do i = jxstart, jxend
      labscb = nint( rout * gasatt )
      mloss(i) = mloss(i) + labscb
   end do
end if

end subroutine xostep
```

A.4.1 Subroutine EXTO

```
! ************************ SUBROUTINE EXTO *************************

! Module Name: EXTO

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine calculates loss based on XO techniques. It
!          performs a ray trace on all rays within one output range step
!          and returns the propagation loss up to the necessary height,
!          storing all angle, height, and range information for ray
```

```
!          trace upon next call.

! Version Number: 1.0

! INPUTS:
!   Argument List: ISTP, JXS, ROUT
!   Common: FTER, HTLIM, IRATZ, ITROPO, IZ, NZOUT
!   Public: CURANG(), CURHT(), CURNG(), FFACZ(,), FFROUT(), FSLR(),
!           GRAD(,), HLIM(), HTR(,), IGRD(), LVL(), ZOUT()

! OUTPUTS:
!   Argument List: JXE, MLOSS()
!   Common: CURANG(), CURHT(), CURNG(), HLIM(), HTOUT(), PRFAC(), RLOSS()

! SAVE: IZE, IZS, IRPS

! Modules Used: APM_MOD

! Calling Routines: XOSTEP

! Routines called:
!   APM Specific: TROPO
!   Intrinsic: AMAX0, AMIN0, AMIN1, NINT

! GLOSSARY: See universal glossary for common variables and parameters.

!   Input Variables:
!       ISTP = index of current output range step.
!       JXS = index in MLOSS() where loss values calculated from
!             XO model begins.
!       ROUT = current output range in meters.

!   Output Variables:
!       JXE = index in MLOSS() where loss values calculated from
!             XO model ends.
!       MLOSS() = 2-byte integer array containing propagation loss values
!                 in centibels vs. height, at each output range ROUT.
!                 All loss values returned are referenced to height HMIN.

!   Local Variables:
!       A0 = Angle at start of trace in radians.
!       A1 = Angle at end of trace in radians.
!       FFAC = Propagation factor in dB for specified output height point
!              at range ROUT.
!       FSLROUT = Free space loss at range ROUT.
!       GRD = Gradient of current refractivity layer being traced through.
!       H0 = Height at start of trace in meters.
!       H1 = Height at end of trace in meters.
!       IGRAD = Index of current gradient level in GRAD(,) in ray trace.
!       IRP = Counter for current refractivity/gradient profile being used
!             from GRAD(,). (Profile varies only for range-dependent case).
!       IRPS = Starting index counter, used to make sure IRP is
!              initialized properly.
!       IZE = Ending index in CURANG(), CURNG(), and CURHT() to trace to
!             ROUT.
!       IZS = Starting index in CURANG(), CURNG(), and CURHT() to trace to
!             ROUT.
!       NXO = # of rays traced, i.e., height points, in XO region.
!       P1 = Propagation factor at height Z1.
!       P2 = Propagation factor at height Z2.
!       R0 = Range at start of trace in meters.
!       R1 = Range at end of trace in meters.
!       Z1 = Nearest traced height point below current output height point
!            in ZOUT().
!       Z2 = Nearest traced height point above current output height point
!            in ZOUT().
```

```
subroutine exto( istp, rout, MLOSS, jxs, JXE )

use apm_mod

integer*2 mloss(0:*)

save ize, izs, irps

! Define in line ray trace functions:

radal( a, b ) = a**2 + 2. * grd * b              !a=a0, b=h1-h0
rp( a, b ) = a + b / grd                         !a=r0, b=a1-a0
ap( a, b ) = a + b * grd                         !a=a0, b=r1-r0
hp( a, b, c ) = a + ( b**2 - c**2 ) / 2. / grd   !a=h0, b=a1, c=a0

! Define in line interpolation function:

plint(pl1, pl2, frac) = pl1 + frac * ( pl2 - pl1 )

! Initialize free space loss.

fslrout = fslr(istp)

! If this is the first time called, then initialize all index variables.

if( istp .eq. iratz ) then
   ize = 1
   izs = 1
   irps = 1
end if

do j = ize, iz
   if( curng(j) .gt. rout ) exit
end do
l = amax0( 1, j-1 )
ize = amin0( l, iz )

k = 0

! Begin trace.

do j = izs, ize
   a0 = curang(j)
   r0 = curng(j)
   h0 = curht(j)
   igrad = igrd(j)

   irp = amax0( j, irps )
   grd = grad(igrad,irp)

   do while ( r0 .lt. rout )

      if( irp .eq. iz ) then
         r1 = rout
      else
         r1 = amin1( ffacz(2,irp+1), rout )
      end if

      a1 = ap( a0, r1-r0 )
      h1 = hp( h0, a1, a0 )

      htrx = htr(igrad+1,irp)
      if( h1 .gt. htrx )then
         h1 = htrx
         rad = radal( a0, h1-h0 )
         a1 = sqrt( rad )
         r1 = rp( r0, a1-a0 )
```

```
            igrad = amin0( igrad+1, lvl(irp)-1 )
         end if

         a0 = a1
         r0 = r1
         h0 = h1

         if( r0 .gt. ffacz(2,irp+1)-1.e-3 ) irp = amin0(irp+1, ize)

      end do

   ! After trace, all angle, range, and height information are stored for
   ! next call to EXTO.  Propagation factor at current range step ROUT, along
   ! with height is stored in PRFAC() and HTOUT(), respectively.

      curht(j) = h0
      curng(j) = r0
      curang(j) = a0
      igrd(j) = igrad
      k = k + 1
      prfac(k) = ffacz(1,j)
      htout(k) = h0
   end do

   irps = ize

   k= k + 1
   prfac(k) = ffrout(1,istp)
   htout(k) = ffrout(2,istp)
   nxo = k

   ! adjust counter of first starting point for ray tracing if ray has
   ! already been traced beyond maximum calculation height.

   do while( curht(izs) .gt. htlim )
      izs = izs + 1
   end do
   izs = amax0( 1, izs - 1 )

   ! Sort height and propagation factor, such that HTOUT() contains steadily
   ! increasing height from HTOUT(NXO) to HTOUT(1).

   if( fter ) then
      k = 1
      do while( k .gt. 0 )
         k = 0
         do j = 1, nxo-1
           if( htout(j) .lt. htout(j+1) ) then
               k = j
               hk = htout(j)
               htout(j) = htout(j+1)
               htout(j+1) = hk
               cf = prfac(j)
               prfac(j) = prfac(j+1)
               prfac(j+1) = cf
            end if
         end do
      end do
   end if

   jxe = nzout
   do while( zout(jxe) .gt. htout(1) )
      jxe = jxe - 1
   end do
   hlim(istp) = zout(jxe)
   ix = nxo
```

```
! Now begin interpolation of propagation factor at specified output
! points ZOUT(i).

z1 = htout(ix)
z2 = htout(ix-1)
p1 = prfac(ix)
p2 = prfac(ix-1)

do j = jxs, jxe
   z = zout(j)

   do while(( z .gt. z2 ) .and. ( ix .gt. 1 ))
      ix = ix - 1
      if( ix .gt. 1 ) then
         z1 = z2
         p1 = p2
         z2 = htout(ix-1)
         p2 = prfac(ix-1)
      end if
   end do

   frac = ( z - z1 ) / ( z2 - z1 )
   ffac = plint( p1, p2, frac )
   rloss(j) = ffac + fslrout
end do

! Compute troposcatter loss and store final loss values in MLOSS().

if( itropo .eq. 1 ) call tropo( istp, jxs, jxe )
do j = jxs, jxe
   mloss(j) = nint( 10. * rloss(j) )
end do

end subroutine exto
```

A.5 Subroutine APMCLEAN

```
!****************************** SUBROUTINE APMCLEAN
******************************

! Module Name: APMCLEAN

! Module Security Classification: UNCLASSIFIED

! Purpose: This routine deallocates all dynamically dimensioned arrays
!          used in one complete run of APM calculations.

! Version Number: 1.0

! INPUTS:
!   Argument List: IERROR
!   Common: ITROPO, NFACS
!   Public: All dynamically dimensioned arrays in APM_MOD

! OUTPUTS:
!   Argument List: IERROR
!   Common: None
!   Public: All dynamically dimensioned (public) arrays.

! Modules Used: APM_MOD

! Calling Routines: MAIN DRIVER PROGRAM

! Routines called:
!   APM Specific: NONE
!   Intrinsic: ALLOCATE, ALLOCATED, DEALLOCATE

! GLOSSARY: See universal glossary for common and public variables.

!   Input Variables:
!            IXOSTP = Index of output range step at which XO model is to
!                     be applied.

!   Output Variables: None
!            IERROR = Integer variable indicating error # for DEALLOCATE
!                     and ALLOCATE statements.

!   Local Variables:
!            NTEMP = Dummy integer variable - used to deallocate FFT arrays
!                    allocated in module SINFFT.

subroutine apmclean( IERROR )

use apm_mod

ierror = 0

! Deallocate all arrays allocated in ALLARRAY_APM.

if( allocated( hfangr ) ) deallocate( hfangr, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rsqrd ) ) deallocate( rsqrd, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( fslr ) ) deallocate( fslr, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rlogo ) ) deallocate( rlogo, stat=ierror )
if( ierror .ne. 0 ) return
```

```
if( allocated( rngout ) ) deallocate( rngout, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( zout ) ) deallocate( zout, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( zro ) ) deallocate( zro, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( zoutma ) ) deallocate( zoutma, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( zoutpa ) ) deallocate( zoutpa, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( hlim ) ) deallocate( hlim, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( htfe ) ) deallocate( htfe, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rfac1 ) ) deallocate( rfac1, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rfac2 ) ) deallocate( rfac2, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rloss ) ) deallocate( rloss, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( tx ) ) deallocate( tx, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( ty ) ) deallocate( ty, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( slp ) ) deallocate( slp, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( dielec ) ) deallocate( dielec, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( igrnd ) ) deallocate( igrnd, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rgrnd ) ) deallocate( rgrnd, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( refdum ) ) deallocate( refdum, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( htdum ) ) deallocate( htdum, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( href ) ) deallocate( href, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( refref ) ) deallocate( refref, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( gr ) ) deallocate( gr, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( q ) ) deallocate( q, stat=ierror )
if( ierror .ne. 0 ) return
```

```
if( allocated( rm ) ) deallocate( rm, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( zrt ) ) deallocate( zrt, stat=ierror )
if( ierror .ne. 0 ) return

! Deallocate arrays used in troposcatter calculations.

if( allocated( adl ) ) deallocate( adl, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( adif ) ) deallocate( adif, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( d2s ) ) deallocate( d2s, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rdt ) ) deallocate( rdt, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( thl ) ) deallocate( thl, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( theta0 ) ) deallocate( theta0, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( theta2s ) ) deallocate( theta2s, stat=ierror )
if( ierror .ne. 0 ) return

! Deallocate all arrays allocated in ALLARRAY_PE.

ntemp = -1
call sinfft( ntemp, xdum )  !Deallocates arrays in SINFFT module.

if( allocated( envpr ) ) deallocate( envpr, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( filt ) ) deallocate( filt, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( frsp ) ) deallocate( frsp, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( ht ) ) deallocate( ht, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( profint ) ) deallocate( profint, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( root ) ) deallocate( root, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( rootm ) ) deallocate( rootm, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( u ) ) deallocate( u, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( ulst ) ) deallocate( ulst, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( w ) ) deallocate( w, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( xdum ) ) deallocate( xdum, stat=ierror )
if( ierror .ne. 0 ) return
```

```
if( allocated( ydum ) ) deallocate( ydum, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( ym ) ) deallocate( ym, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( cn2 ) ) deallocate( cn2, stat=ierror )
if( ierror .ne. 0 ) return

! Deallocate all arrays allocated in ALLARRAY_XO.

if( allocated( ffrout ) ) deallocate( ffrout, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( ffacz ) ) deallocate( ffacz, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( grad ) ) deallocate( grad, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( htr ) ) deallocate( htr, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( lvl ) ) deallocate( lvl, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( filtp ) ) deallocate( filtp, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( xp ) ) deallocate( xp, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( yp ) ) deallocate( yp, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( spectr ) ) deallocate( spectr, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( curang ) ) deallocate( curang, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( curht ) ) deallocate( curht, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( curng ) ) deallocate( curng, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( igrd ) ) deallocate( igrd, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( htout ) ) deallocate( htout, stat=ierror )
if( ierror .ne. 0 ) return

if( allocated( prfac ) ) deallocate( prfac, stat=ierror )
if( ierror .ne. 0 ) return

end subroutine apmclean
```

A.6 MODULE APM_MOD

```
module apm_mod

   implicit integer*4 (i-n)

   parameter ( pi = 3.1415926 )   !Self-explanatory
   parameter ( irtemp = 200 )     !Number of range steps for use in ray-tracing
                                  !to determine maximum PE angle.

! ERRORFLAG:
!  LERR6 =  Logical flag that allows for greater flexibility in allowing error
!           -6 to be bypassed.  If set to .TRUE. then trapping for this error
!           occurs, otherwise it can be totally ignored by main driver
!           program. (Within the APM program it is handled as a warning).  If
!           this error is bypassed (LERR6 = .FALSE.) terrain profile is
!           extended to RMAX with same elevation height of last valid terrain
!           profile point.
!  LERR12 = Same as LERR6 - allows for trapping of this error.  If LERR12 =
!           .FALSE., then (for range-dependent case) if range of last
!           refractivity profile entered is less than RMAX, the environment
!           is treated as homogeneous from the last profile entered to RMAX.

   common / errorflag / lerr6, lerr12
   logical lerr6, lerr12

! INPUTVAR:
!   HMAX = maximum output height with respect to m.s.l. in meters
!   HMIN = minimum output height with respect to m.s.l. in meters
!   ITROPO = integer flag indicating if troposcatter solutions are
!            to be performed. ITROPO=0 -> no troposcatter calculations,
!            ITROPO=1 -> perform troposcatter calculations.
!   NZOUT = integer number of output height points desired
!   NROUT = integer number of output range points desired
!   RMAX = maximum output range in meters

   common / inputvar / hmax, hmin, itropo, nzout, nrout, rmax
   real hmax, hmin, rmax

! REFRACTIVITY common block and associated input variables:
!   ABSHUM = absolute humidity near the surface in g/m3.
!   GAMMAA = gaseous absorption in dB/km.
!   HMSL(,) = Dynamically allocated 2-dimensional array of size
!             (0:LVLP,NPROF) containing heights in meters with respect
!             to mean sea level of each profile.  Array format must be
!             HMSL(I,J) = height of Ith level of Jth profile.  J = 1
!             for range-independent cases.

!             ************************** NOTE: **************************
!             LVLP is the actual # of height levels occupying 0 to LVLP-1
!             elements in array HMSL; there will be an extra point
!             unused on input.
!             ***********************************************************

!   IEXTRA = Extrapolation flag for refractivity profiles entered below
!            m.s.l.
!            IEXTRA = 0 -> extrapolate to minimum terrain height using
!                     standard atmosphere gradient.
!            IEXTRA = 1 -> extrapolate to minimum terrain height using
!                     first gradient in profile.
!   LVLP = number of levels in refractivity profile (for range dependent
!          case all profiles must have same number of levels).
!   NPROF = number of profiles.  Equals 1 for range-independent cases.
!   REFMSL(,) = Dynamically allocated 2-dimensional array of size
!               (0:LVLP,NPROF) containing refractivity with respect to
```

```
!                mean sea level of each profile.  Array format must be
!                REFMSL(I,J) = M-unit value at Ith level of Jth profile.
!                J = 1 for range-independent cases.

!              ************************* NOTE: *************************
!              LVLP is the actual # of refractivity levels occupying 0 to
!              LVLP-1 elements in array REFMSL; there will be an extra
!              point unused on input.
!              ********************************************************

!   RNGPROF() = ranges of each profile in meters, i.e., RNGPROF(I) = range of
!                Ith profile.  RNGPROF(1) should always be equal to 0.
!   TAIR = air temperature near the surface in degrees C.

!****************************** NOTE *********************************
! For DEC Visual Fortran Ver. 5.0 Fortran 90 compilation using
! dynamically allocated arrays, the following source code MUST be in the
! main driver (calling) program before initialization of the arrays can
! can be performed:

! IF( ALLOCATED( HMSL ) ) DEALLOCATE( HMSL )
! ALLOCATE( HMSL(0:LVLP, NPROF) )
! HMSL = 0.

! IF( ALLOCATED( REFMSL ) ) DEALLOCATE( REFMSL )
! ALLOCATE( REFMSL(0:LVLP, NPROF) )
! REFMSL = 0.

! IF( ALLOCATED( RNGPROF ) ) DEALLOCATE( RNGPROF )
! ALLOCATE( RNGPROF(NPROF) )
! RNGPROF = 0.

! Once the above source code has been inserted in the main (calling)
! routine, the arrays can then be initialized with the desired refrac-
! tivity profiles for subsequent use by routines APMINIT and APMSTEP.
!*********************************************************************

   common / refractivity / abshum, gammaa, iextra, lvlp, nprof, tair

   real tair, abshum, gammaa
   real, allocatable :: hmsl(:,:), refmsl(:,:), rngprof(:)
   public :: hmsl, refmsl, rngprof

! SYSTEMVAR:
!   ANTHT = transmitting antenna height above local ground in meters.
!   BWIDTH = half-power (3 dB) antenna pattern beamwidth in degrees (.5 to 45.)
!   ELEV = antenna pattern elevation angle in degrees. (-10 to 10)
!   FREQ = frequency in MHz
!   HFANG() = Dynamically allocated array of cut-back angles in degrees.
!             This is only used for user-defined height-finder antenna type.
!   HFFAC() = Dynamically allocated array of cut-back antenna pattern
!             factors. This is only used for user-defined height-finder
!             antenna type.
!   IPAT = integer value indicating type of antenna pattern desired
!          IPAT = 1 -> omni
!          IPAT = 2 -> gaussian
!          IPAT = 3 -> sinc x
!          IPAT = 4 -> csc**2 x
!          IPAT = 5 -> generic height-finder
!          IPAT = 6 -> user-defined height-finder
!   IPOL = integer indicating polarization.  0-horizontal,
!          1-vertical
!   NFACS = Number of user-defined cut-back angles and cut-back antenna
!           pattern factors for user-defined height-finder antenna type.

!****************************** NOTE *********************************
! For DEC Visual Fortran Ver. 5.0 Fortran 90 compilation using
```

```
! dynamically allocated arrays, the following source code MUST be in the
! main driver (calling) program before initialization of the arrays can
! can be performed:

! IF( ALLOCATED( HFANG ) ) DEALLOCATE( HFANG )
! ALLOCATE( HFANG(NFACS) )
! HFANG = 0.

! IF( ALLOCATED( HFFAC ) ) DEALLOCATE( HFFAC )
! ALLOCATE( HFFAC(NFACS) )
! HFFAC = 0.

! Once the above source code has been inserted in the main (calling)
! routine, the arrays can then be initialized with the desired cut-back
! angles and factors for subsequent use by routines APMINIT and APMSTEP.
!***********************************************************************

  common / systemvar / antht, bwidth, elev, freq, ipat, ipol, nfacs

  real freq, antht, bwidth, elev
  real, allocatable :: hfang(:), hffac(:)
  public :: hfang, hffac

! TERRAIN common block and associated input variables:
!  DIELEC(,) = Dynamically allocated 2-dimensional array of size (2,IGR)
!              containing the relative permittivity and conductivity;
!              DIELEC(1,i) and DIELEC(2,i), respectively. Only needs to be
!              specified if using IGRND(i) = 7, otherwise, APM will
!              calculate based on frequency and ground types 0-6.
!  IGR = number of different ground types specified
!  IGRND() = Dynamically allocated integer array of size (IGR) containing
!            ground type composition for given terrain profile - can
!            vary with range. Different ground types are:
!               0 = sea water
!               1 = fresh water
!               2 = wet ground
!               3 = medium dry ground
!               4 = very dry ground
!               5 = ice at -1 degree C
!               6 = ice at -10 degree C
!               7 = user defined (in which case, values of relative
!                   permittivity and conductivity must be given).
!  ITP = number of height/range pairs in profile
!  RGRND() = Dynamically allocated array of size (IGR) containing ranges,
!            in m, at which the ground types apply.
!  TERX() = Dynamically allocated array of size (ITP) containing range
!           points of terrain profile in meters.
!  TERY() = Dynamically allocated array of size (ITP) containing height
!           points of terrain profile in meters.

!****************************** NOTE **********************************
! For DEC Visual Fortran Ver. 5.0 Fortran 90 compilation using
! dynamically allocated arrays, the following source code MUST be in the
! main driver (calling) program before initialization of the arrays can
! can be performed:

! IF( ALLOCATED( DIELEC ) ) DEALLOCATE( DIELEC )
! ALLOCATE( DIELEC(2, IGR) )
! DIELEC = 0.

! IF( ALLOCATED( IGRND ) ) DEALLOCATE( IGRND )
! ALLOCATE( IGRND(IGR) )
! IGRND = 0.

! IF( ALLOCATED( RGRND ) ) DEALLOCATE( RGRND )
! ALLOCATE( RGRND(IGR) )
! RGRND = 0.
```

```
! IF( ALLOCATED( TERX ) ) DEALLOCATE( TERX )
! ALLOCATE( TERX(ITP) )
! TERX = 0.

! IF( ALLOCATED( TERY ) ) DEALLOCATE( TERY )
! ALLOCATE( TERY(ITP) )
! TERY = 0.

! Once the above source code has been inserted in the main (calling)
! routine, the arrays can then be initialized with the desired terrain
! information for subsequent use by routines APMINIT and APMSTEP.
!***********************************************************************

   common / terrain / igr, itp

   real, allocatable :: dielec(:,:), rgrnd(:), terx(:), tery(:)
   allocatable :: igrnd(:)
   public :: dielec, igrnd, rgrnd, terx, tery

!****************** START OF INTERNAL APM DECLARATIONS *****************

! Common Blocks

! ABSORB:
!   GASATT = Gaseous absorption in dB/km.
!   KABS = Integer flag indicating whether or not to compute gaseous
!          absorption loss.  KABS=0 no absorption loss; KABS=1 compute
!          absorption loss based on air temperature TAIR and absolute
!          humidity ABSHUM; KABS=2 compute absorption loss based on specified
!          absorption attenuation rate GAMMAA.

! IMPEDANCE:
!   ALPHAV = vertical polarization impedance term = i*FKO/RNG.
!   C1 = Coefficient used in vertical polarization calculations.
!   C2 = Coefficient used in vertical polarization calculations.
!   C1X = Constant dependent on each new calculated RT - used to
!         calculate C1 at next range step.
!   C2X = Constant dependent on each new calculated RT - used to
!         calculate C2 at next range step.
!   IG = Counter indicating current ground type being modeled.
!   RK = Coefficient used in C1 and C2 calculations.
!   RT = complex root of quadratic equation for mixed transform method
!        based on Kuttler's formulation.

! MISCVAR:
!   AEK2 = 2. * AEK
!   ALPHAD = Direct ray elevation angle in radians
!   ANTREF = transmitting antenna height relative to the reference
!            height HMINTER.
!   FKO = free-space wavenumber = (2*pi) / WL
!   FKO2 = 2. * FKO
!   FTER = logical flag - .TRUE.=terrain case, .FALSE.=smooth surface case
!   HMREF = height relative to HMINTER.  Determined from user-provided
!           minimum height HMIN. That is, if HMIN is minimum height input
!           by user with respect to mean sea level, and HMINTER is
!           internally considered the new origin, then HMREF = HMIN - HMINTER.
!   HTLIM = user-supplied maximum height relative to HMINTER, i.e.,
!         HTLIM = HMAX - HMINTER
!   IHYBRID = Integer indicating which sub-models will be used:
!             0 = pure PE model
!             1 = full hybrid model
!             2 = PE + XO model
!   ITPA = Number of terrain points used internally in arrays TX() and TY().
!   IXO =  Number of range steps in XO calculation region.
!   IZG = Output height integer index indicating the start of good loss
!         values (in PE region) for a particular output range.
```

```
!   PLCNST = constant used in determining propagation loss
!            PLCNST = 20log(2*FKO).
!   RHOR = Radar horizon range in meters for 0 receiver height.
!   RLOG = 10. * alog10( PE range )
!   RLOGLST = RLOG of previous range step (i.e., 10*alog10(PE range-DR) )
!   RPEST = Range in meters at which loss values from the PE model will
!           start being calculated.
!   TWOKA = Twice the effective earth's radius factor times the effective
!           earth radius.  The effective earth's radius factor is calcula-
!           ted based on a ray trace at 5 degrees from the origin to the
!           HTLIM.  This is used for routine FEM.
!   WL = Wavelength in meters
!   YCUR = height of ground at the current range step
!   YCURM = height of ground midway between last and current range step.
!           For use when shifting profiles to be relative to the local ground
!           height.
!   YFREF = Ground elevation height at source.
!   YLAST = height of ground at the last range step

! OUTRH:
!   DROUT = Output range step in meters
!   DZOUT = Output height increment in meters

! PATTERN:
!   AFAC = constant used in determining antenna pattern factors
!          AFAC = 1.39157 / sin( bw / 2 ) for SIN(X)/X and height-finder
!          AFAC = (.5*ln(2))/(sin(bw/2))**2 for GAUSSIAN
!   BW = antenna pattern beamwidth in radians
!   ELV = antenna pattern elevation angle in radians
!   PELEV = sine of elevation angle
!   SBW = sine of the beamwidth
!   UMAX = limiting angle used in cut-off point for SIN(X)/X and
!          generic height-finder antenna pattern factors

! PE:
!   CNST = used in calculating ENVPR() in routine PHASE1.
!          CNST = DELP/FKO.
!   CON = 1.e-6 * FKO; Constant used in calculation of ENVPR()
!   DELP = mesh size in angle- (or p-) space.
!   DELZ = Bin width in z-space = WL / (2*sin(THETAMAX))
!   DR = PE range step in meters
!   DR2 = 1/2 PE range step in meters
!   DZ2 = 2. * DELZ
!   FNORM = normalization factor used for DFT.
!   LN = Power of 2 transform size, i.e. N = 2**LN
!   LNMIN = Minimum power of 2 transform size.  LNMIN = 9 for smooth
!           surface and frequencies <= 3000 MHz.  LNMIN = 10 all other
!           cases.
!   N = Transform size
!   N34 = 3/4 * N
!   N4 = N / 4
!   NM1 = N-1
!   THETA75 = 75% of maximum propagation angle in PE calculations.
!   ZLIM = Maximum internal height (HTLIM) or .75*ZMAX, whichever is smaller.
!   ZMAX = Maximum height of PE calculation domain = N * DELZ

! REFPROF:
!   HMINTER = Minimum height of terrain profile in meters.  This will be
!             used to adjust entire terrain profile so all internal
!             calculations will be referenced to this height.
!   IS = counter for current profile (for range-dependent cases)
!   LVLEP = Number of height/refractivity levels in profile REFDUM(),HTDUM()
!           taken w.r.t. reference height HMINTER.
!   NLVL = Number of height/refractivity levels in profile REFREF(),HREF()
!          taken w.r.t. local ground height at middle of range step, YCURM.
!   RV2 = range of the next refractivity profile (for range-dependent cases)
```

```
! RO:
!   DELXRO = RO range increment in meters
!   DMAGSQ(,) = direct-ray magnitude squared
!   HTYDIF = height difference between internal maximum height, HTLIM,
!            and initial ground height at the source, YFREF.
!   IRON = next index for RO solution (0 or 1)
!   IROP = previous index for RO solution (-1, 0, or 1)
!   ISTART= RO height index at transmitter
!   KMAX = maximum K-index at XROP and XRON
!   KMINN = minimum K-index at XRON
!   KMINP = minimum K-index at XROP
!   LEVELS = number of levels defined in ZRT(), Q(), and GR() arrays
!   OMEGA(,) = phase angle between direct & reflected rays in radians
!   PSILIM = grazing angle of limiting ray in radians
!   RMAGSQ(,) = reflected-ray magnitude squared
!   XLIMRO = range of limiting ray in meters
!   XREFLECT = Range at which ray is reflected in RO and FE calculations.
!   XRON = next range for RO solution in meters
!   XROP = previous range for RO solution in meters
!   ZTOL = height tolerance for Newton's method in meters

! SPEC:
!   LNP = Power of 2 transform size used in spectral estimation calcs.
!   NP34 = 3/4 * NPNTS
!   NPNTS = Number of points used in top part of PE region for spectral
!           estimation.
!   NS = Transform size used in spectral estimation calcs = 2**LNP
!   XOCON = Constant used in determining outgoing propagation angle
!           for XO calcs -> WL / (2*NS*DELZ).

! TROPOV:
!   JT1 = Index counter for AD1() and TH1() arrays.
!   JT2 = Index counter for TX() and TY() arrays indicating where receiver
!         range is, i.e., TX(JT2-1) < ROUT < TX(JT2).
!   KTR1 = Number of increasing tangent angles and ranges determined from
!          source height over terrain path profile.
!   R1T = Constant used in troposcatter calcs. = RF*ANTREF
!   RF = Constant used in troposcatter calcs. = 4*PI*FREQ/speed of light
!        (x10e6)
!   SN1 = Term used in troposcatter loss calc.
!   THETA1S = Tangent angle from source for smooth surface.
!   TLSTS = Troposcatter loss term for smooth surface (non-terrain).

! TRVAR:
!   AATZ = local propagation angle at height ZLIM and range RATZ
!          (used for hybrid model).
!   ALAUNCH = Launch angle used, in radians, which, when traced, separates
!             PE & XO regions from RO region
!   HTEMP() = Heights at which ray is traced to every range point RTEMP(i)
!   IAP = Index indicating when local ray angle becomes positive in array
!         RAYA().
!   IRATZ = Index of output range step at which ZLIM is reached (for
!           hybrid model only).  Indicates at what range step begin
!           storing propagation factor and outgoing angle for XO region.
!   RATZ = Range at which ZLIM is reached (used for hybrid model).
!   RAYA() = Array containing all local angles of traced ray ALAUNCH at
!            each output range.
!   RTEMP() = Range steps for tracing to determine maximum PE angle.

! The XO common block is only used when using hybrid model.
! XO:
!   IZ = Counter for points stored in FFACZ(,) array.
!   IZINC = Integer increment for storing points at top of PE region to
!           start XO model.  I.E., points are stored at every IZINC range
!           step.
!      IZMAX = Maximum # of points allocated for arrays associated with
!              XO calcs.
```

```
!   JZLIM = PE bin # corresponding to ZLIM, i.e., ZLIM = JZLIM*DELZ.

   common / absorb / gasatt, kabs

   common / impedance / alphav, c1, c2, c1x, c2x, ig, rk, rt

   common / miscvar / aek2, alphad, antref, fko, fko2, fter, hmref, htlim,&
                      ihybrid, itpa, izg, plcnst, rhor, rlog, rloglst, &
                      rpest, twoka, wl, ycur, ycurm, yfref, ylast, ixo

   common / outrh / drout, dzout

   common / pattern / afac, bw, elv, pelev, sbw, umax

   common / pe / cnst, con, delp, delz, dr, dr2, dz2, fnorm, ln, lnmin,  &
                 n, n34, n4, nm1, theta75, zlim, zmax

   common / refprof / hminter, is, lvlep, nlvl, rv2

   common / ro / delxRO, dmagsq(0:1,0:88), htydif, iROp, iROn, istart,    &
                 kminn, kminp, kmax, levels, omega(0:1,0:88), psilim,     &
                 rmagsq(0:1,0:88), xlimRO, xreflect, xROn, xROp, ztol

   common / spec / lnp, np34, npnts, ns, xocon, np4

   common / tropov / jt1, jt2, ktr1, r1t, rf, sn1, theta1s, tlsts

   common / trvar / aatz, alaunch, htemp(irtemp), iap, iratz, ratz,      &
                    raya(irtemp), rtemp(irtemp)

   common / xo / iz, izinc, izmax, jzlim

!****************** DYNAMICALLY ALLOCATED ARRAYS ************************

!   AD1() = Tangent ranges from source height w/ terrain path profile.
!   ADIF() = Height array in meters used for troposcatter calcs.
!   CN2() = Complex dielectric constant.
!   CURANG() = Current local angle for each ray being traced in XO region.
!   CURHT() = Current local height for each ray being traced in XO region.
!   CURNG() = Current local range for each ray being traced in XO region
!   D2S() = Tangent range array in meters for all output receiver heights
!           over smooth surface.
!   ENVPR() = Complex array containing refractivity exponential term.
!             i.e. ENVPR() = exp[i * DR * FKO * 1e-6 * M(z) ], where
!             M(z) is the refractivity at each PE bin height z.
!   FFACZ(,) = 2-dimensional array containing propagation factor in dB,
!              range, and propagation angle at ZLIM.  Used to start XO
!              calculations.
!              FFACZ(1,IZ) = propagation factor in dB at current PE range
!              FFACZ(2,IZ) = current PE range
!              FFACZ(3,IZ) = propagation angle at current PE range at ZLIM
!   FFROUT() = Propagation factor in dB at each output range step
!              beyond RATZ and at height ZLIM
!   FILT() = Cosine-tapered (Tukey) filter array.
!   FILTP() = Array filter for spectral estimation calcs.
!   FRSP() = Complex array containing free-space propagator exponential term.
!            i.e., FRSP() = exp[-i * DR * (FKO - sqrt(FKO**2 - p**2)) ]
!   FSLR() = array containing the free space loss at every output range.
!   GR() = 1.E-6 * dM/dz array used for RO calculations
!   GRAD(,) = 2-dimensional array containing gradients of each refrac-
!             tivity profile vs. range from height ZLIM to HTLIM.
!   HFANGR() = array of user-defined cut-back angles in radians.  This is
!              used only for user-defined height-finder antenna type.
!   HLIM() = array containing height at each output range separating the
!            RO region from the PE (at close ranges) and XO (at farther
!            ranges) regions
!   HREF() = Heights of refractivity profile with respect to YREF (local
```

```
!              ground height).
!    HT() = Height array of size N.  Heights space every DELZ.
!    HTDUM() = Height array containing height values for current (interpolated)
!               profile in meters, relative to HMINTER.
!    HTFE() = Array containing the height at each output range step separa-
!              ting the flat earth region from the RO region.
!    HTOUT() = Final height for each ray traced in XO region at range
!               ROUT.
!    HTR(,) = 2-dimensional array containing height levels of each refrac-
!              tivity profile vs. range from height ZLIM to HTLIM.
!    IGRD() = Integer indexes indicating at what gradient in GRAD(,) to
!              begin raytracing for next XO range step for each ray in XO
!              region.
!    LVL() = Number of refractivity levels in current refractivity profile
!             from ZLIM to HTLIM.
!    PRFAC() = Propagation factor for each ray traced in XO region at
!               range ROUT.
!    PROFINT() = M-unit profile interpolated to every DELZ in height
!    Q() = 2 * [RM(i+1)-RM(i)] array used for RO calculations
!    RDT() = Minimum range array (in meters) at which diffraction field
!             solutions are applicable and intermediate region ends (for
!             smooth surface) for all output receiver heights.
!    REFDUM() = dummy array containing M-unit values for current (interpolated)
!                profile taken relative to HMINTER.
!    REFREF() = Refractivity array w.r.t. YREF (local ground height).
!    RFAC1() = Propagation factor at valid output height points computed
!               from PE field at previous PE range, i.e., ULST().
!    RFAC2() = Propagation factor at valid output height points computed
!               from PE field at current PE range, i.e., U().
!    RLOGO() = Array of logarithm of output ranges, i.e., RLOGO(i) =
!               20. * ALOG10(i*DROUT).
!    RLOSS() = Propagation loss in dB.
!    RM() = 1.E-6 * M array used for RO calculations
!    RNGOUT() = array containing all output ranges in meters.
!    ROOT() = array of RT to the i'th power, i.e. ROOT(I) = RT**I
!    ROOTM() = array of -RT to the i'th power, i.e. ROOTM(I) = (-RT)**I
!    RSQRD() = double precision array containing the square of output ranges
!    SLP() = slope of each segment of terrain.
!    SPECTR() = Field amplitude of spectral portion of PE field in dB.
!    TH1() = Tangent angles from source height w/ terrain path profile.
!    THETA0() = Angle array - angles used in determining common volume
!                scattering angle.
!    THETA2S() = Tangent angle array from all output receiver heights for
!                 smooth surface.
!    TX() = range points of terrain profile in meters.
!    TY() = adjusted height points of terrain profile in meters.
!    U() = Complex array containing PE field solution.
!    ULST() = Complex array containg PE field solution at previous range step.
!    W() = Difference equation of complex PE field array.  Used in
!           intermediate calculations only for vertical polarization.
!    XDUM() = Real part of complex PE field array U().
!    XP() = Real part of spectral portion of PE field.
!    YDUM() = Imaginary part of complex PE field array U().
!    YM() = Particular solution of difference equation.  Used in
!            intermediate calculations only for vertical polarization.
!    YP() = Imaginary part of spectral portion of PE field.
!    ZOUT() = array containing all output heights in meters referenced to
!              HMINTER.
!    ZOUTMA() = Array containing output heights in meters relative to the
!                antenna height above ground at 0 range.  Used in FE model.
!    ZOUTPA() = Array containing output heights in meters relative to the
!                image antenna height below ground at 0 range.  Used in FE
!                model.
!    ZRO() = output height array in meters referenced to ground elevation
!             height at source.  Used for RO calculations.
!    ZRT() = height array used for RO calculations in meters
```

```
   complex, allocatable :: envpr(:), frsp(:), root(:), rootm(:), u(:),    &
                           ulst(:), w(:), ym(:), cn2(:)
   public :: envpr, frsp, root, rootm, u, ulst, w, ym, cn2

   integer, allocatable :: igrd(:), lvl(:)
   public :: igrd, lvl

   double precision, allocatable :: rsqrd(:)
   public :: rsqrd

   real, allocatable :: adl(:), adif(:), curang(:), curht(:), curng(:),  &
                        d2s(:), ffacz(:,:), ffrout(:,:), filt(:),         &
                        filtp(:), fslr(:), gr(:), grad(:,:), hfangr(:),  &
                        hlim(:), href(:), ht(:), htdum(:), htfe(:),       &
                        htout(:), htr(:,:), prfac(:), profint(:), q(:),  &
                        rdt(:), refdum(:), refref(:), rfac1(:), rfac2(:),&
                        rlogo(:), rloss(:), rm(:), rngout(:), slp(:),    &
                        spectr(:), thl(:), theta0(:), theta2s(:), tx(:), &
                        ty(:), xdum(:), xp(:), ydum(:), yp(:), zout(:),  &
                        zoutma(:), zoutpa(:), zro(:), zrt(:)

    public :: adl, adif, curang, curht, curng, d2s, ffacz, ffrout,   &
              filt, filtp, fslr, gr, grad, hfangr, hlim, href,       &
              ht, htdum, htfe, htout, htr, prfac,  profint, q,       &
              rdt, refdum, refref, rfac1, rfac2, rloss, rlogo, rm,   &
              rngout, slp, spectr, thl, theta0, theta2s, tx, ty,     &
              xdum, xp, ydum, yp, zout, zoutma, zoutpa, zro, zrt


   complex alphav, c2x, rk, clx, cl, c2, rt

   logical fter

   data aek / 8.4946667e6 /    !4/3 times mean earth radius in m
   data ek / 1.3333333 /       !4/3 effective earth's radius factor
   data radc / 1.74533e-2 /    !degree to radian conversion factor
   data rtst / 2500. /         !Range set at 2.5 km to begin calculation
                               !  of RO values.

contains
```

SINFFT subroutine (refer to Section 8.1.18)

```
end module apm_mod
```

SOFTWARE TEST DESCRIPTION

FOR THE

ADVANCED PROPAGATION MODEL CSCI

(Version 1.0)

August 1998

Prepared for:

Space and Naval Warfare Systems Command (PMW-185)

San Diego, CA

Prepared by:

Space and Naval Warfare Systems Center San Diego

Tropospheric Branch (Code D883)

49170 Propagation Path

San Diego, CA 92152-7385

CONTENTS

CONTENTS (Continued)

Tables

1. SCOPE

1.1 IDENTIFICATION

The Advanced Propagation Model (APM) Version 1.0 computer software configuration item (CSCI) calculates range-dependent electromagnetic (EM) system propagation loss within a heterogeneous atmospheric medium over variable terrain, where the radio-frequency index of refraction is allowed to vary both vertically and horizontally. Numerous Tactical Environmental Support System-Next Century (TESS-NC) applications require EM-system propagation loss values. The APM model described by this document may be applied to two such TESS-NC applications, one that displays propagation loss on a range versus height scale (commonly referred to as a coverage diagram) and one that displays propagation loss on a propagation loss versus range/height scale (commonly referred to as a loss diagram).

1.2 DOCUMENT OVERVIEW

This document specifies the test cases and test procedures necessary to perform qualification testing of the APM CSCI. A discussion of precise input values of each input variable required to perform the test together with final expected test results is presented.

2. REFERENCE DOCUMENTS

(a) Commander-In-Chief, Pacific Fleet Meteorological Requirement (PAC MET) 87-04, "Range Dependent Electromagnetic Propagation Models."

(b) Naval Oceanographic Office. 1990. "Software Documentation Standards and Coding Requirements for Environmental System Product Development," Apr.

(c) Naval Command, Control and Ocean Surveillance Center; Research, Development, Test and Evaluation Division (NRaD), 1997. "Terrain Parabolic Equation Model (TPEM) Computer Software Configuration Item (CSCI) Documents." NRaD TD 2963 (May) San Diego, CA.

(d) Naval Command, Control and Ocean Surveillance Center; Research, Development, Test and Evaluation Division (NRaD). 1997. "Radio Physical Optics (RPO) CSCI Software Documents, RPO Ver. 1.16", NRaD TD 2403 Rev. 1 (Apr), San Diego, CA.

(e) Space and Naval Warfare (SPAWAR) Systems Center, San Diego (SSC San Diego). 1998. "Software Requirements Specification for the Advanced Propagation Model (APM) CSCI," Aug.

(f) Space and Naval Warfare (SPWAR) Systems Center, San Diego (SSC San Diego). 1998. "Software Design Document for the Advanced Propagation Model (APM) CSCI,"Aug.

(g) Barrios, A. E., "Terrain Parabolic Equation Model (TPEM) Version 1.5 User's Manual," Naval Command, Control and Ocean Surveillance Center RDT&E Division, San Diego, CA, NRaD TD 2898, February 1996.

3. TEST PREPARATIONS

3.1 HARDWARE PREPARATION

Not applicable

3.2 SOFTWARE PREPARATION

A short driver program, APMMAIN.F90, is provided in Section 7. This program exercises the main software components, APMINIT CSC, APMSTEP CSC, XOINIT CSC, XOSTEP CSC, and APMCLEAN CSC that comprise the APM CSCI. The driver program demonstrates how to access the APM CSCI and to exercise the test cases listed in the following sections. It is written to read all necessary input data for the test cases from files in a specific format. All necessary input information is presented in tabular form in Section 4.3 and the input files for each test case are listed in Section 8.

One of the main features of APM is the use of dynamic allocation in most of the arrays used for both numeric calculations and as inputs to the model. Care must be taken by the TESS-NC CSCI application designer to properly allocate memory and initialize all variable and array inputs to APM. Ultimately, it is the responsibility of the TESS-NC CSCI application designer to provide the necessary input in the form required by the APM CSCI.

3.2 OTHER PRETEST PREPARATION

None.

4. TEST DESCRIPTIONS

The test specification for the APM CSCI consists of 28 separate tests that exercise all subroutines and functions of the CSCI. For ease of testing, each of these 28 tests is given a name describing which portion of the APM CSCI is being exercised. All 28 tests and their descriptions are listed in table 1.

Table 1. Test names and descriptions.

Test Name	Description
ABSORB	Gaseous absorption attenuation rate is specified.
BLOCK	The terrain profile consists of a vertical flat-topped block or obstacle in which the terrain slope is undefined.
COSEC2	Antenna pattern is of cosecant-squared type.
EDUCT	The refractivity consists of a 14 meter evaporation duct profile.
GASABS	The surface absolute humidity and surface air temperature are specified in order to compute a gaseous absorption attenuation rate.
GAUSS	Antenna pattern is of Gaussian type.
HIBW	Large vertical beamwidth is specified.
HIEL	High elevation angle is specified.

Table 1. Test names and descriptions. (Continued)

Test Name	Description
HIFREQ	High frequency.
HITRAN	High transmitter antenna height.
HORZ	Horizontal polarization antenna and standard atmosphere.
HTFIND	Antenna pattern is of generic height-finder type.
LOBW	Small vertical beamwidth is specified.
LOEL	Low elevation angle is specified.
LOFREQ	Low frequency.
LOTRAN	Low transmitter antenna height.
RDLONGB	Range-dependent refractivity over a DTED-extracted terrain profile from Long Beach to Point Mugu, using vertical polarization and generic ground composition types.
RNGDEP	Range-dependent refractivity over smooth earth (over-water case).
SBDUCT	300 meter surface-based duct.
SINEX	Antenna pattern is of Sine(X)/X type.
TROPOS	Troposcatter for smooth surface (over-water case).
TROPOT	Troposcatter over terrain.
USERHF	Antenna pattern is of specific height finder type, with user-specified cut-back angles and power factors.
VERT	Vertical polarization antenna is specified (short range over-water case, standard atmosphere).
VERTMIX	Vertical polarization antenna over mixed land-sea terrain path.
VERTSEA	Vertical polarization antenna is specified (long range over-water case, ducting atmosphere).
VERTUSRD	Vertical polarization antenna and user-specified dielectric ground constants.
WEDGE	The terrain profile consists of a triangular wedge.

4.1 REQUIREMENTS ADDRESSED

Not applicable.

4.2 PREREQUISITE CONDITIONS

None.

4.3 TEST INPUTS

Although there are actual values for all input parameters listed in the input files in Section 8, some are ignored depending on the values of certain input parameters. Those input parameters that are inapplicable depending on the test case are listed as "N/A" in the tables. Note that for all test cases, the error flags, *lerr6* and *lerr12*, are set to ".TRUE.". These flags allow for extra error control regarding terrain and refractivity inputs. We recommend that these error flags always be set to ".TRUE.". However, we allowed the capability of the TESS-NC applications designer to bypass these error controls according to the application.

The external environmental data element requirements are listed in table 2 for each test name, with tables 3 through 7 providing specific height and M-unit values. The external EM system data element requirements are listed in table 8.

Table 2. External environmental data element requirements.[a]

Test Name	$hmsl$ Table	$refmsl$ Table	n_{prof}	$lvlp$	$rngprof$ Table	abs_{hum} g/m^3	t_{air} °C	γ_a dB/km
ABSORB	3	3	1	2	0.	0.	0.	.146
BLOCK	3	3	1	2	0.	0.	0.	0.
COSEC2	3	3	1	2	0.	0.	0.	0.
EDUCT	5	5	1	21	0.	0.	0.	0.
GASABS	3	3	1	2	0.	10.	25.	0.
GAUSS	3	3	1	2	0.	0.	0.	0.
HIBW	3	3	1	2	0.	0.	0.	0.
HIEL	3	3	1	2	0.	0.	0.	0.
HIFREQ	3	3	1	2	0.	0.	0.	0.
HITRAN	3	3	1	2	0.	0.	0.	0.
HORZ	3	3	1	2	0.	0.	0.	0.
HTFIND	3	3	1	2	0.	0.	0.	0.
LOBW	3	3	1	2	0.	0.	0.	0.
LOEL	3	3	1	2	0.	0.	0.	0.
LOFREQ	3	3	1	2	0.	0.	0.	0.
LOTRAN	3	3	1	2	0.	0.	0.	0.
RDLONGB	6	6	2	4	6	0.	0.	0.
RNGDEP	7	7	2	4	7	0.	0.	0.
SBDUCT	4	4	1	4	0.	0.	0.	0.
SINEX	3	3	1	2	0.	0.	0.	0.
TROPOS	3	3	1	2	0.	0.	0.	0.
TROPOT	3	3	1	2	0.	0.	0.	0.
USERHF	3	3	1	2	0.	0.	0.	0.
VERT	3	3	1	2	0.	0.	0.	0.
VERTMIX	3	3	1	2	0.	0.	0.	0.
VERTSEA	4	4	1	4	0.	0.	0.	0.
VERTUSRD	3	3	1	2	0.	0.	0.	0.
WEDGE	3	3	1	2	0.	0.	0.	0.

[a]The interpolation flag, i_{extra}, is set to 0 for all test cases.

Table 3. Standard atmosphere with 118-M/km gradient.

i	$hmsl_{i,1}$ (meters)	$refmsl_{i,1}$ (M-unit)
1	0	350
2	1000	468

Table 4. 300-meter surface-based duct atmosphere.

i	$hmsl_{i,1}$ (meters)	$refmsl_{i,1}$ (M-unit)
1	0.0	339.0
2	250.0	368.5
3	300.0	319.0
4	1000.0	401.6

Table 5. Atmosphere with 14-meter evaporation duct.

I	$hmsl_{i,1}$ (meters)	$refmsl_{i,1}$ (M-unit)
1	0.000	339.00
2	0.040	335.10
3	0.100	333.66
4	0.200	332.60
5	0.398	331.54
6	0.794	330.51
7	1.585	329.53
8	4.362	328.65
9	6.310	327.96
10	12.589	327.68
11	14.000	327.67
12	25.119	328.13
13	39.811	329.25
14	50.119	330.18
15	63.096	331.44
16	79.433	334.32
17	100.000	335.33
18	125.893	338.20
19	158.489	341.92
20	199.526	346.69
21	209.526	347.87

Table 6. Range-dependent atmosphere, standard atmosphere to surface-based duct.

	Standard Atmosphere $rngprof_1 = 0$ km		Surface-based Duct $rngprof_2 = 100$ km	
i	$hmsl_{i,1}$ (meters)	$refmsl_{i,1}$ (M-unit)	$hmsl_{i,2}$ (meters)	$refmsl_{i,2}$ (M-unit)
1	0.0	350.0	0.0	339.0
2	0.0	350.0	250.0	368.5
3	0.0	350.0	300.0	319.0
4	1000.0	468.0	1000.0	401.6

Table 7. Range-dependent atmosphere, surface-based duct to high elevated duct.

	Surface-based Duct $rngprof_1 = 0.$ Km		High Elevated Duct $rngprof_2 = 250.$ km	
i	$hmsl_{i,1}$ (meters)	$refmsl_{i,1}$ (M-unit)	$hmsl_{i,2}$ (meters)	$refmsl_{i,2}$ (M-unit)
1	0.0	330.0	0.0	330.0
2	100.0	342.5	600.0	405.0
3	230.0	312.5	730.0	375.0
4	2000.0	517.8	2000.0	522.3

Table 8. External EM System data element requirements.

Test Name	f_{MHz} (MHz)	ant_{ht} (meters)	i_{pat} note a	i_{pol} note b	μ_{bw} (deg)	μ_o (deg)	n_{facs}	$hfang$ (deg)	$hffac$
ABSORB	20000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
BLOCK	1000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
COSEC2	1000.0	25.	4	0	1.	0.	N/A	N/A	N/A
EDUCT	10000.0	15.	2	0	5.	0.	N/A	N/A	N/A
GASABS	20000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
GAUSS	1000.0	25.	2	0	1.	0.	N/A	N/A	N/A
HIBW	1000.0	25.	2	0	45.	0.	N/A	N/A	N/A
HIEL	1000.0	25	2	0	1.	10.	N/A	N/A	N/A
HIFREQ	20000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
HITRAN	1000.0	100.	1	0	N/A	N/A	N/A	N/A	N/A
HORZ	1000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
HTFIND	1000.0	25.	5	0	2.	0.	N/A	N/A	N/A
LOBW	1000.0	25.	2	0	.5	0.	N/A	N/A	N/A
LOEL	1000.0	25.	2	0	1.	-10.	N/A	N/A	N/A

Table 8. External EM System data element requirements. (Continued)

Test Name	f_{MHz} (MHz)	ant_{ht} (meters)	i_{pat} note a	i_{pol} note b	μ_{bw} (deg)	μ_o (deg)	n_{facs}	*hfang* (deg)	*hffac*
LOFREQ	100.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
LOTRAN	1000.0	1.	1	0	N/A	N/A	N/A	N/A	N/A
RDLONGB	150.0	100.	1	0	N/A	N/A	N/A	N/A	N/A
RNGDEP	3000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
SBDUCT	3000.0	25.	2	0	5.	0.	N/A	N/A	N/A
SINEX	1000.0	25.	3	0	1.	0.	N/A	N/A	N/A
TROPOS	100.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
TROPOT	100.0	25.	1	0	N/A	N/A	N/A	N/A	N/A
USERHF	1000.0	25.	6	0	1.	0.	10	note c	note d
VERT	1000.0	25.	1	1	N/A	N/A	N/A	N/A	N/A
VERTMIX	100.0	10.	1	1	N/A	N/A	N/A	N/A	N/A
VERTSEA	100.0	25.	1	1	N/A	N/A	N/A	N/A	N/A
VERTUSRD	100.0	10.	1	1	N/A	N/A	N/A	N/A	N/A
WEDGE	1000.0	25.	1	0	N/A	N/A	N/A	N/A	N/A

[a]Antenna Pattern: 1 = Omni-directional; 2 = Gaussian; 3 = Sine(X)/X; 4 = Cosecant-squared; 5 = Generic height-finder; 6 = User-specified height finder.

[b]Polarization: 0 = Horizontal; 1 = Vertical

[c]Power reduction angles (*hfang*): 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5

[d]Power reduction factors (*hffac*): 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0

The external implementation data element requirements that must be specified for each test are listed in table 9.

Table 9. External implementation data element requirements.

Test Name	*lerr6*	*lerr12*	n_{rout}	n_{zout}	r_{max} (meters)	h_{min} (meters)	h_{max} (meters)	i_{tropo}
ABSORB	.true.	.true.	1	20	50,000.0	0.0	200.0	0
BLOCK	.true.	.true.	1	20	50,000.0	0.0	1000.0	0
COSEC2	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
EDUCT	.true.	.true.	1	20	50,000.0	0.0	200.0	0
GASABS	.true.	.true.	1	20	50,000.0	0.0	200.0	0
GAUSS	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
HIBW	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
HIEL	.true.	.true.	1	20	50,000.0	0.0	20,000.0	0

Table 9. External implementation data element requirements. (Continuted)

Test Name	*lerr6*	*lerr12*	n_{rout}	n_{zout}	r_{max} (meters)	h_{min} (meters)	h_{max} (meters)	i_{tropo}
HIFREQ	.true.	.true.	1	20	50,000.0	0.0	200.	0
HITRAN	.true.	.true.	1	20	50,000.0	0.0	1000.0	0
HORZ	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
HTFIND	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
LOBW	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
LOEL	.true.	.true.	1	20	50,000.0	0.0	20,000.0	0
LOFREQ	.true.	.true.	1	20	50,000.0	0.0	5000.0	0
LOTRAN	.true.	.true.	1	20	50,000.0	0.0	10,000.0	0
RDLONGB	.true.	.true.	1	20	100,000.0	0.0	1000.0	0
RNGDEP	.true.	.true.	1	20	250,000.0	0.0	2000.0	0
SBDUCT	.true.	.true.	1	20	200,000.0	0.0	5000.0	0
SINEX	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
TROPOS	.true.	.true.	1	20	200,000.0	0.0	2000.0	1
TROPOT	.true.	.true.	1	20	200,000.0	0.0	2000.0	1
USERHF	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
VERT	.true.	.true.	1	20	50,000.0	0.0	2000.0	0
VERTMIX	.true.	.true.	1	20	50,000.0	0.0	1000.0	0
VERTSEA	.true.	.true.	1	20	300,000.0	0.0	1000.0	0
VERTUSRD	.true.	.true.	1	20	50,000.0	0.0	1000.0	0
WEDGE	.true.	.true.	1	20	100,000.0	0.0	1000.0	0

The external terrain data element requirements are listed in table 10. Terrain profiles used for specific test cases are listed in tables 11 through 15.

Table 10. External terrain data element requirements.

Test Name	*terx* Table	*tery* Table	i_{tp}	i_{gr}	*igrnd* Table	*rgrnd* Table	*Dielec* (ε_r,σ)[a]
ABSORB	N/A	N/A	N/A	N/A	N/A	N/A	N/A
BLOCK	11	11	6	N/A	N/A	N/A	N/A
COSEC2	N/A	N/A	N/A	N/A	N/A	N/A	N/A
EDUCT	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GASABS	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GAUSS	N/A	N/A	N/A	N/A	N/A	N/A	N/A
HIBW	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 10. External terrain data element requirements. (Continued)

Test Name	*terx* Table	*tery* Table	i_{tp}	I_{gr}	*igrnd* Table	*rgrnd* Table	*dielec* (ε,σ)[a]
HIEL	N/A	N/A	N/A	N/A	N/A	N/A	
HIFREQ	N/A	N/A	N/A	N/A	N/A	N/A	N/A
HITRAN	N/A	N/A	N/A	N/A	N/A	N/A	N/A
HORZ	N/A	N/A	N/A	N/A	N/A	N/A	N/A
HTFIND	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LOBW	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LOEL	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LOFREQ	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LOTRAN	N/A	N/A	N/A	N/A	N/A	N/A	N/A
RDLONGB	4-12	4-12	167	6	4-13	4-13	N/A
RNGDEP	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SBDUCT	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SINEX	N/A	N/A	N/A	N/A	N/A	N/A	N/A
TROPOS	N/A	N/A	N/A	N/A	N/A	N/A	N/A
TROPOT	12	12	167	6	13	13	N/A
USERHF	N/A	N/A	N/A	N/A	N/A	N/A	N/A
VERT	N/A	N/A	N/A	1	0	0.	N/A
VERTMIX	14	14	2	2	14	14	N/A
VERTSEA	N/A	N/A	N/A	1	0	0.	N/A
VERTUSRD	N/A	N/A	N/A	1	7	0.	3., 6e-4
WEDGE	15	15	5	N/A	N/A	N/A	N/A

[a] ε_r = relative permittivity; σ = conductivity (S/m)

Table 11. Terrain profile for Test Case BLOCK.

i	$terx_i$ (meters)	$tery_i$ (meters)
1	0.0	0.0
2	22,500.0	0.0
3	22,500.0	200.0
4	27,500.0	200.0
5	27,500.0	0.0
6	50,000.0	0.0

Table 12. Terrain profile for Test Case RDLONGB.

i	$terx_i$ (meters)	$tery_i$ (meters)	i	$terx_i$ (meters)	$tery_i$ (meters)	i	$terx_i$ (meters)	$tery_i$ (meters)
1	0.0	8.0	57	20100.	22.0	113	79200.0	184.0
2	300.0	8.0	58	20400.	23.0	114	79500.0	226.0
3	600.0	9.0	59	20700.	24.0	115	79800.0	152.0
4	900.0	9.0	60	21000.	24.0	116	80100.0	201.0
5	1200.0	10.0	61	21300.	25.0	117	80400.0	244.0
6	1500.0	11.0	62	21600.	26.0	118	80700.0	152.0
7	1800.0	12.0	63	21900.	27.0	119	81000.0	143.0
8	2100.0	13.0	64	22200.	27.0	120	81300.0	91.0
9	2400.0	14.0	65	22500.	28.0	121	81600.0	107.0
10	2700.0	15.0	66	22800.	29.0	122	81900.0	152.0
11	3000.0	17.0	67	23400.	29.0	123	82200.0	152.0
12	3300.0	19.0	68	23700.	30.0	124	82500.0	170.0
13	3600.0	21.0	69	24600.	30.0	125	82800.0	152.0
14	3900.0	23.0	70	24900.	32.0	126	83100.0	66.0
15	4200.0	25.0	71	25200.	34.0	127	83400.0	70.0
16	4500.0	27.0	72	25500.	38.0	128	83700.0	121.0
17	4800.0	28.0	73	26100.	38.0	129	84000.0	152.0
18	5100.0	30.0	74	26400.	36.0	130	84300.0	170.0
19	5400.0	31.0	75	26700.	34.0	131	84600.0	141.0
20	5700.0	31.0	76	27000.	32.0	132	84900.0	139.0
21	6000.0	29.0	77	27300.	27.0	133	85200.0	147.0
22	6300.0	23.0	78	27600.	15.0	134	85500.0	177.0
23	6600.0	14.0	79	27900.	6.0	135	85800.0	152.0
24	6900.0	9.0	80	28200.	1.0	136	86100.0	61.0
25	7200.0	7.0	81	28500.	0.0	137	86700.0	61.0
26	7500.0	7.0	82	64500.	0.0	138	87000.0	70.0
27	7800.0	9.0	83	64800.	8.0	139	87300.0	44.0
28	8100.0	11.0	84	65100.	30.0	140	87600.0	11.0
29	8400.0	14.0	85	65400.	39.0	141	87900.0	1.0
30	8700.0	13.0	86	65700.	61.0	142	89400.0	1.0
31	9300.0	13.0	87	66600.	61.0	143	89700.0	61.0
32	9600.0	12.0	88	66900.	24.0	144	90000.0	84.0
33	9900.0	11.0	89	67200.	14.0	145	90300.0	152.0
34	10200.0	8.0	90	67500.	26.0	146	90600.0	152.0
35	10800.0	8.0	91	67800.	16.0	147	90900.0	101.0
36	11100.0	7.0	92	68100.	1.0	148	91200.0	40.0
37	12600.0	7.0	93	68400.	1.0	149	91500.0	15.0
38	12900.0	6.0	94	68700.	0.0	150	91800.0	20.0
39	14400.0	6.0	95	73800.	0.0	151	92100.0	2.0
40	14700.0	7.0	96	74100.	1.0	152	92400.0	10.0
41	15000.0	8.0	97	74400.	1.0	153	92700.0	4.0

Table 12. Terrain profile for Test Case RDLONGB. (Continued)

i	$terx_i$ (meters)	$tery_i$ (meters)	i	$terx_i$ (meters)	$tery_i$ (meters)	i	$terx_i$ (meters)	$tery_i$ (meters)
42	15300.0	8.0	98	74700.	10.0	154	93000.0	1.0
43	15600.0	9.0	99	75000.0	8.0	155	93300.0	1.0
44	15900.0	10.0	100	75300.0	39.0	156	93600.0	0.0
45	16200.0	11.0	101	75600.0	45.0	157	93900.0	1.0
46	16500.0	11.0	102	75900.0	53.0	158	96300.0	1.0
47	16800.0	12.0	103	76200.0	61.0	159	96600.0	0.0
48	17400.0	12.0	104	76500.0	61.0	160	96900.0	1.0
49	17700.0	13.0	105	76800.0	82.0	161	97500.0	1.0
50	18000.0	13.0	106	77100.0	61.0	162	97800.0	2.0
51	18300.0	14.0	107	77400.0	78.0	163	98100.0	3.0
52	18600.0	15.0	108	77700.0	61.0	164	99300.0	3.0
53	18900.0	16.0	109	78000.0	129.0	165	99600.0	2.0
54	19200.0	18.0	110	78300.0	30.0	166	99900.0	2.0
55	19500.0	20.0	111	78600.0	46.0	167	100200.0	1.0
56	19800.0	21.0	112	78900.0	159.0			

Table 13. Table of ground constants for terrain profile of table 12.

i_{gr}	$igrnd_i$ (Note a)	$rgrnd_i$ (meters)
1	2	0
2	0	28,500
3	3	64,800
4	0	68,700
5	4	74,100
6	0	100,200

[a]Ground composition type: 0 = sea water; 1 = fresh water; 2 = wet ground; 3 = medium dry ground; 4 = very dry ground; 5 = ice at -1 °C; 6 = ice at -10 °C; 7 = user-defined permittivity and conductivity.

Table 14. Terrain profile for Test Case VERTMIX.

i	$terx_i$ (meters)	$tery_i$ (meters)	i_{gr}	$igrnd_i$ (Note a)	$rgrnd_i$ (meters)
1	0.	0.	1	4	0.
2	50,000.	0.	2	0	25000.

[a]Ground composition type: 0 = sea water; 1 = fresh water; 2 = wet ground; 3 = medium dry ground; 4 = very dry ground; 5 = ice at -1 °C; 6 = ice at -10 °C; 7 = user-defined permittivity and conductivity.

Table 15. Terrain profile for Test Case WEDGE.

i	$terx_i$ (meters)	$tery_i$ (meters)
1	0.0	0.0
2	45000.0	0.0
3	50000.0	200.0
4	55000.0	0.0
5	100000.0	0.0

4.4 EXPECTED TEST RESULTS

The expected test result propagation loss versus height values for each of the 28 test cases are listed in tabular form in tables 16 through 43.

Table 16. Expected Output for ABSORB Test.

Height (meters)	Propagation Loss (dB)
10.0	212.9
20.0	199.3
30.0	188.9
40.0	180.1
50.0	172.2
60.0	165.5
70.0	160.1
80.0	156.7
90.0	156.5
100.0	163.2
110.0	159.3
120.0	156.0
130.0	167.8
140.0	155.7
150.0	163.0
160.0	156.1
170.0	161.9
180.0	155.7
190.0	164.5
200.0	154.9

Table 17. Expected output for BLOCK Test.

Height (meters)	Propagation Loss (dB)
50.0	173.6
100.0	170.1
150.0	167.0
200.0	162.4
250.0	157.0
300.0	151.4
350.0	145.8
400.0	140.3
450.0	135.0
500.0	129.6
550.0	124.2
600.0	120.5
650.0	121.0
700.0	128.1
750.0	141.3
800.0	125.3
850.0	121.3
900.0	120.5
950.0	122.1
1000.0	127.5

Table 18. Expected output for COSEC2 Test

Height (meters)	Propagation Loss (dB)
100.0	134.4
200.0	124.1
300.0	122.3
400.0	129.7
500.0	126.5
600.0	123.5
700.0	128.0
800.0	126.9
900.0	125.7
1000.0	126.6
1100.0	127.1
1200.0	127.5
1300.0	128.9
1400.0	129.5
1500.0	129.7
1600.0	131.0
1700.0	131.5
1800.0	131.4
1900.0	132.7
2000.0	133.1

Table 19. Expected output for EDUCT Test.

Height (meters)	Propagation Loss (dB)
10.0	142.8
20.0	147.5
30.0	150.1
40.0	152.5
50.0	156.0
60.0	158.6
70.0	154.1
80.0	149.5
90.0	146.3
100.0	144.3
110.0	143.1
120.0	142.7
130.0	143.2
140.0	145.1
150.0	149.5
160.0	161.7
170.0	151.9
180.0	145.2
190.0	142.4
200.0	141.5

Table 20. Expected output for GASABS Test.

Height (meters)	Propagation Loss (dB)
10.0	212.9
20.0	199.3
30.0	188.9
40.0	180.1
50.0	172.2
60.0	165.5
70.0	160.1
80.0	156.7
90.0	156.5
100.0	163.2
110.0	159.3
120.0	156.0
130.0	167.8
140.0	155.7
150.0	163.0
160.0	156.1
170.0	161.9
180.0	155.7
190.0	164.5
200.0	154.9

Table 21. Expected output for GAUSS Test.

Height (meters)	Propagation Loss (dB)
100.0	133.7
200.0	123.5
300.0	121.7
400.0	130.7
500.0	127.1
600.0	124.0
700.0	133.0
800.0	132.2
900.0	129.6
1000.0	139.0
1100.0	140.0
1200.0	138.1
1300.0	148.3
1400.0	150.4
1500.0	149.5
1600.0	160.5
1700.0	163.6
1800.0	163.7
1900.0	175.4
2000.0	179.6

Table 22. Expected output for HIBW Test.

Height (meters)	Propagation Loss (dB)
100.0	133.6
200.0	123.3
300.0	121.1
400.0	129.7
500.0	124.9
600.0	120.6
700.0	128.1
800.0	125.3
900.0	120.5
1000.0	127.5
1100.0	125.6
1200.0	120.5
1300.0	127.5
1400.0	125.6
1500.0	120.5
1600.0	127.4
1700.0	125.7
1800.0	120.5
1900.0	127.4
2000.0	125.7

Table 23. Expected output for HIEL Test.

Height (meters)	Propagation Loss (dB)
1000.0	376.4
2000.0	376.4
3000.0	376.4
4000.0	376.4
5000.0	364.2
6000.0	258.9
7000.0	184.7
8000.0	140.8
9000.0	126.6
10000.0	141.2
11000.0	183.7
12000.0	253.1
13000.0	348.2
14000.0	376.7
15000.0	376.8
16000.0	376.8
17000.0	376.9
18000.0	376.9
19000.0	377.0
20000.0	377.1

Table 24. Expected output for HIFREQ Test.

Height (meters)	Propagation Loss (dB)
10.0	205.6
20.0	192.0
30.0	181.6
40.0	172.8
50.0	164.9
60.0	158.2
70.0	152.8
80.0	149.4
90.0	149.2
100.0	155.9
110.0	152.0
120.0	148.7
130.0	160.5
140.0	148.4
150.0	155.7
160.0	148.8
170.0	154.6
180.0	148.4
190.0	157.2
200.0	147.6

Table 25. Expected output for HITRAN Test.

Height (meters)	Propagation Loss (dB)
50.0	126.3
100.0	121.8
150.0	138.1
200.0	121.4
250.0	134.6
300.0	122.0
350.0	124.4
400.0	127.7
450.0	120.9
500.0	131.4
550.0	123.2
600.0	121.5
650.0	148.1
700.0	121.7
750.0	122.3
800.0	137.7
850.0	121.1
900.0	123.2
950.0	132.4
1000.0	120.8

Table 26. Expected output for HORZ Test.

Height (meters)	Propagation Loss (dB)
100.0	133.6
200.0	123.3
300.0	121.1
400.0	129.7
500.0	124.9
600.0	120.6
700.0	128.1
800.0	125.3
900.0	120.5
1000.0	127.5
1100.0	125.6
1200.0	120.4
1300.0	127.4
1400.0	125.6
1500.0	120.4
1600.0	127.4
1700.0	125.6
1800.0	120.4
1900.0	127.4
2000.0	125.6

Table 27. Expected output for HTFIND Test.

Height (meters)	Propagation Loss (dB)
100.0	133.6
200.0	123.4
300.0	121.5
400.0	130.0
500.0	125.8
600.0	122.1
700.0	128.7
800.0	127.0
900.0	124.4
1000.0	127.1
1100.0	126.5
1200.0	126.4
1300.0	126.4
1400.0	126.4
1500.0	126.4
1600.0	126.4
1700.0	126.4
1800.0	126.4
1900.0	126.4
2000.0	126.4

Table 28. Expected output for LOBW Test.

Height (meters)	Propagation Loss (dB)
100.0	133.8
200.0	124.0
300.0	123.2
400.0	132.9
500.0	133.0
600.0	134.1
700.0	146.4
800.0	151.8
900.0	156.6
1000.0	171.5
1100.0	181.5
1200.0	190.4
1300.0	208.4
1400.0	222.4
1500.0	235.6
1600.0	256.6
1700.0	274.6
1800.0	292.0
1900.0	316.2
2000.0	338.1

Table 29. Expected output for LOEL Test.

Height (meters)	Propagation Loss (dB)
1000.0	376.4
2000.0	376.4
3000.0	376.4
4000.0	376.4
5000.0	358.2
6000.0	254.5
7000.0	181.7
8000.0	139.4
9000.0	126.6
10000.0	142.7
11000.0	186.5
12000.0	257.2
13000.0	353.6
14000.0	376.7
15000.0	376.8
16000.0	376.8
17000.0	376.9
18000.0	376.9
19000.0	377.0
20000.0	377.1

Table 30. Expected output for LOFREQ Test.

Height (meters)	Propagation Loss (dB)
250.0	116.7
500.0	109.0
750.0	105.0
1000.0	102.6
1250.0	101.2
1500.0	100.5
1750.0	100.5
2000.0	101.0
2250.0	102.3
2500.0	104.5
2750.0	108.4
3000.0	117.0
3250.0	119.5
3500.0	109.2
3750.0	104.9
4000.0	102.6
4250.0	101.2
4500.0	100.6
4750.0	100.5
5000.0	101.0

Table 31. Expected output for LOTRAN Test.

Height (meters)	Propagation Loss (dB)
500.0	137.0
1000.0	129.5
1500.0	125.8
2000.0	123.5
2500.0	122.0
3000.0	121.0
3500.0	120.5
4000.0	120.4
4500.0	120.7
5000.0	121.4
5500.0	122.5
6000.0	124.2
6500.0	126.9
7000.0	131.2
7500.0	141.2
8000.0	139.7
8500.0	130.8
9000.0	126.7
9500.0	124.3
10000.0	122.6

Table 32. Expected output for RDLONGB Test.

Height (meters)	Propagation Loss (dB)
50.0	145.3
100.0	139.0
150.0	134.3
200.0	131.2
250.0	131.9
300.0	133.8
350.0	128.5
400.0	125.7
450.0	123.0
500.0	121.8
550.0	121.3
600.0	120.5
650.0	118.5
700.0	115.8
750.0	115.5
800.0	115.7
850.0	113.2
900.0	112.7
950.0	111.9
1000.0	111.7

Table 33. Expected output for RNGDEP Test.

Height (meters)	Propagation Loss (dB)
100.0	199.7
200.0	195.7
300.0	202.5
400.0	178.6
500.0	141.9
600.0	135.4
700.0	150.9
800.0	164.2
900.0	166.8
1000.0	182.9
1100.0	196.6
1200.0	197.5
1300.0	200.5
1400.0	195.1
1500.0	193.1
1600.0	191.7
1700.0	192.2
1800.0	194.2
1900.0	196.3
2000.0	193.0

Table 34. Expected output for SBDUCT Test.

Height (meters)	Propagation Loss (dB)
250.00	139.8
500.00	166.0
750.00	157.0
1000.0	161.1
1250.0	176.5
1500.0	167.1
1750.0	158.4
2000.0	150.7
2250.0	146.5
2500.0	146.7
2750.0	163.0
3000.0	144.7
3250.0	147.1
3500.0	147.0
3750.0	145.3
4000.0	149.7
4250.0	144.3
4500.0	149.9
4750.0	144.5
5000.0	148.0

Table 35. Expected output for SINEX Test.

Height (meters)	Propagation Loss (dB)
100.0	133.7
200.0	123.5
300.0	121.6
400.0	130.7
500.0	127.0
600.0	124.1
700.0	133.3
800.0	133.0
900.0	131.9
1000.0	143.2
1100.0	150.6
1200.0	376.4
1300.0	376.4
1400.0	376.4
1500.0	376.4
1600.0	376.4
1700.0	376.4
1800.0	376.4
1900.0	376.4
2000.0	376.4

Table 36. Expected output for TROPOS Test.

Height (meters)	Propagation Loss (dB)
100.0	165.2
200.0	164.5
300.0	164.5
400.0	164.4
500.0	164.4
600.0	164.2
700.0	163.4
800.0	162.1
900.0	161.2
1000.0	158.5
1100.0	155.9
1200.0	153.5
1300.0	151.2
1400.0	149.0
1500.0	146.9
1600.0	144.9
1700.0	143.1
1800.0	141.3
1900.0	139.6
2000.0	138.0

Table 37. Expected output for TROPOT Test.

Height (meters)	Propagation Loss (dB)
100.0	164.8
200.0	163.8
300.0	162.8
400.0	161.3
500.0	159.3
600.0	157.3
700.0	155.4
800.0	154.0
900.0	153.0
1000.0	152.3
1100.0	151.9
1200.0	151.2
1300.0	149.9
1400.0	148.0
1500.0	146.0
1600.0	144.1
1700.0	142.4
1800.0	140.9
1900.0	139.3
2000.0	137.6

Table 38. Expected output for USERHF Test.

Height (meters)	Propagation Loss (dB)
100.00	133.7
200.00	123.8
300.00	122.6
400.00	128.9
500.00	126.7
600.00	126.0
700.00	126.4
800.00	126.4
900.00	126.4
1000.0	126.4
1100.0	127.3
1200.0	127.3
1300.0	127.3
1400.0	127.3
1500.0	128.4
1600.0	128.4
1700.0	128.4
1800.0	128.4
1900.0	128.4
2000.0	129.5

Table 39. Expected output for VERT Test.

Height (meters)	Propagation Loss (dB)
100.0	133.8
200.0	123.5
300.0	121.4
400.0	129.3
500.0	125.9
600.0	121.3
700.0	127.8
800.0	127.0
900.0	121.7
1000.0	127.2
1100.0	127.9
1200.0	122.1
1300.0	126.9
1400.0	128.5
1500.0	122.5
1600.0	126.7
1700.0	128.9
1800.0	122.8
1900.0	126.5
2000.0	129.1

Table 40. Expected output for VERTMIX Test.

Height (meters)	Propagation Loss (dB)
50.0	142.2
100.0	135.2
150.0	130.9
200.0	127.7
250.0	125.1
300.0	122.9
350.0	121.0
400.0	119.3
450.0	118.0
500.0	116.8
550.0	115.8
600.0	114.9
650.0	114.1
700.0	113.3
750.0	112.6
800.0	112.0
850.0	111.4
900.0	110.8
950.0	110.3
1000.0	109.8

Table 41. Expected output for VERTSEA Test.

Height (meters)	Propagation Loss (dB)
50.0	136.5
100.0	130.0
150.0	127.2
200.0	126.8
250.0	129.3
300.0	136.0
350.0	143.9
400.0	147.8
450.0	146.5
500.0	145.1
550.0	144.3
600.0	144.1
650.0	144.2
700.0	144.3
750.0	144.7
800.0	145.1
850.0	145.5
900.0	146.0
950.0	146.3
1000.0	146.7

Table 42. Expected output for VERTUSRD Test.

Height (meters)	Propagation Loss (dB)
50.00	141.0
100.0	134.4
150.0	130.3
200.0	127.2
250.0	124.7
300.0	122.7
350.0	120.9
400.0	119.4
450.0	118.1
500.0	116.9
550.0	115.9
600.0	114.9
650.0	114.1
700.0	113.3
750.0	112.6
800.0	112.0
850.0	111.4
900.0	110.8
950.0	110.3
1000.	109.8

Table 43. Expected output for WEDGE Test.

Height (meters)	Propagation Loss (dB)
50.0	157.6
100.0	156.4
150.0	155.8
200.0	155.0
250.0	154.6
300.0	154.9
350.0	155.1
400.0	152.7
450.0	149.2
500.0	146.7
550.0	144.6
600.0	141.4
650.0	137.2
700.0	133.2
750.0	129.5
800.0	126.7
850.0	126.0
900.0	127.9
950.0	127.8
1000.0	129.5

4.5 CRITERIA FOR EVALUATING RESULTS

The calculated propagation loss in dB should match the numerical values in each table at each of the 20 levels shown to within 0.1 dB (1 cB). APM rounds its output loss values to the nearest 1 cB, and hence it is possible for differences of 1 cB to exist between different implementations of APM. It is expected, however, that in most cases the values will match those in tables 16 through 43 exactly.

4.6 TEST PROCEDURE

1. Compile for execution, the APM CSCI, the driver program APMMAIN.F90, and the module APM_MOD.F90.
2. An input data file has been provided, as a text file, for each test case.
3. The APM CSCI is executed in a form that reads the input data file, performs the calculations, and writes the output to a text file.
4. The output file is compared to the final expected test results to determine satisfactory performance.

4.7 ASSUMPTIONS AND CONSTRAINTS

Input data elements are assumed to be constrained by the limits listed within Tables 1 through 4 of the Software Requirements Specification, reference e.

5. REQUIREMENTS TRACEABILITY

1. The provided driver program that accesses the APM CSCI will create an output file for each test case. The output file will have the same prefix name as the input file. The extension is ".OUT". This output file contains height in meters and corresponding propagation loss in dB that should correspond to the entries in tables 16 through 43 for each test case.

2. The provided program APMMAIN.FOR, when compiled with the APM CSCI, will read the provided input files containing all necessary information for each test case. Each input file is named for each test case, with a ".IN" extension.

6. NOTES

Table 44 is a glossary of acronyms and abbreviations used within this document.

Table 44. Acronyms and abbreviations.

Term	Definition
abs_{hum}	Surface absolute humidity (g/m^3)
ant_{ht}	Antenna height
APM	Advanced Propagation Model
μ_{bw}	Antenna vertical beam width (degrees)
cB	Centibel
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
dB	Decibel
dielec	Two-dimensional array of relative permittivity and conductivity
μ_o	Antenna elevation angle (degrees)
EM	Electromagnetic
FORTRAN	Formula Translation
f_{MHz}	EM system frequency (MHz)
γ_a	Surface specific attenuation rate (dB/km)
hfang	User-defined height-finder power reduction angle array (deg)
hffac	User-defined power reduction factor array
h_{max}	Maximum height output for a particular application of APM.

Table 44. Acronyms and abbreviations. (Continued)

Term	Definition
h_{min}	Minimum height output for a particular application of APM.
hmsl	refractivity profile height array
i_{extra}	Extrapolation flag for refractivity profiles entered below mean sea level
i_{gr}	Number of ground composition types for particular application of APM
igrnd	Ground composition type array
i_{pat}	Antenna pattern
i_{pol}	Antenna polarization
i_{tp}	Number of terrain points for particular application of APM
i_{tropo}	Flag to include troposcatter calculations
lerr6	Controlling logical flag for error -6
lerr12	Controlling logical flag for error -12
lvlp	Number of levels in refractivity profiles for particular application of APM
km	Kilometers
m	Meters
N/A	Not applicable
n_{facs}	Number of power reduction factors and cut-back angles for user-defined height finder radar
n_{prof}	Number of refractivity profiles for particular application of APM
n_{rout}	Number of range output points for a particular application of APM.
n_{zout}	Number of height output points for a particular application of APM.
refmsl	Refractivity profile M-unit array
rgrnd	Round composition type range array
r_{max}	Maximum range output for a particular application of APM.

Table 44. Acronyms and abbreviations. (Continued)

Term	Definition
rngprof	Refractivity profile range array
t_{air}	Surface air temperature (°C)
terx	Terrain profile range array
tery	Terrain profile height array
TESS-NC	Tactical Environmental Support System-Next Century

7. SAMPLE PROGRAM LISTING

The sample driver program APMMAIN.FOR, to exercise the APM CSCI is provided below.

```
! This is a sample driver program for APM routines APMINIT, APMSTEP,
! XOINIT, and XOSTEP. All numeric parameters passed to APMINIT and
! APMSTEP must be in metric units. All input arrays are dynamically
! allocated and are dimensioned with variable sizes.

program apmmain

use apm_mod

!MLOSS must be declared an INTEGER*2 allocatable array.
!ITLOSS is a dummy array and will be used to store entire loss grid.

integer*2, allocatable :: mloss(:), itloss(:,:)
character filein*20, fileout*24, answer*1

10 continue

write(*,'(a\)')' Name of input file? '
read(*, '(a)' ) filein

open(14, file=filein)

!*********************************READ SYSTEM INFO******************************

read( 14, * ) lerr6
read( 14, * ) lerr12

read( 14, * ) freq    !Frequency in MHz.
read( 14, * ) antht    !antenna height.
read( 14, * ) ipat    !antenna type
read( 14, * ) ipol    !antenna polarization.
read( 14, * ) bwidth   !This value is ignored for Omni antenna, otherwise,
             !the value must be entered in degrees.
read( 14, * ) elev    !This value is ignored for Omni antenna, otherwise,
```

```
            !the value must be entered in degrees.
read( 14, * ) nfacs    !If using specific height-finder antenna, this variable
            !contains a non-zero value corresponding to the # of
            !cut-back angles and cut-back factors.

! If using specific height-finder antenna, then must specify values for HFANG()
! and HFFAC arrays. Height-finder cut-back angles HFANG() must be in degrees.

if( nfacs .gt. 0 ) then
  IF( ALLOCATED( hfang ) ) DEALLOCATE( hfang, stat=ierror )
  ALLOCATE( hfang(nfacs), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN HFANG ALLOCATION***********'
   stop
  end if
  hfang = 0.

  IF( ALLOCATED( hffac ) ) DEALLOCATE( hffac, stat=ierror )
  ALLOCATE( hffac(nfacs), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN HFFAC ALLOCATION***********'
   stop
  end if
  hffac = 0.

  read( 14, * )( hfang(i), i=1, nfacs )
  read( 14, * )( hffac(i), i=1, nfacs )
end if

!***************************READ GENERIC INPUT INFO******************************

read( 14, * ) hmin    !Minimum height in m
read( 14, * ) hmax    !Maximum output height in m
read( 14, * ) rmax    !Maximum output range in m
read( 14, * ) nzout    !Number of output height points.
read( 14, * ) nrout    !Number of output range points.
read( 14, * ) itropo   !Troposcatter flag: 0=no troposcatter, 1=troposcatter

!Allocate and initialize MLOSS() and ITLOSS() arrays.

if( allocated( mloss ) ) deallocate( mloss, stat=ierror )
```

```
allocate( mloss(0:nzout), stat = ierror )
if( ierror .ne. 0 ) then
  write(*,*)'******ERROR IN MLOSS ALLOCATION*******'
  stop
end if
mloss = 0.

if( allocated( itloss ) ) deallocate( itloss, stat=ierror )
allocate( itloss(nrout,0:nzout), stat=ierror )
if( ierror .ne. 0 ) then
  write(*,*)'******ERROR IN ITLOSS ALLOCATION*******'
  stop
end if
itloss = 0.

!***************************READ METEOROLOGICAL INFO****************************

read( 14, * ) nprof    !Number of refractivity profiles
read( 14, * ) lvlp     !Number of levels in refractivity profiles.
read( 14, * ) iextra   !Extrapolation flag: 0=extrapolate using standard
             !gradient,1=extrapolate using gradient from first 2
             !levels.
read( 14, * ) abshum   !Surface absolute humidity in g/m**3
read( 14, * ) tair     !Surface air temperature in degrees
read( 14, * ) gammaa   !Gaseous absorption attenuation rate in dB/km

! Allocate and initialize height/refractivity and range arrays.

IF( ALLOCATED( HMSL ) ) DEALLOCATE( HMSL, stat=ierror )
ALLOCATE( HMSL(0:LVLP, NPROF), stat=ierror )
if( ierror .ne. 0 ) then
  write(*,*)'**********ERROR IN HMSL ALLOCATION***********'
  stop
end if
HMSL = 0.

IF( ALLOCATED( REFMSL ) ) DEALLOCATE( REFMSL, stat=ierror )
ALLOCATE( REFMSL(0:LVLP, NPROF), stat=ierror )
if( ierror .ne. 0 ) then
  write(*,*)'**********ERROR IN REFMSL ALLOCATION***********'
  stop
```

```
end if
REFMSL = 0.

IF( ALLOCATED( RNGPROF ) ) DEALLOCATE( RNGPROF, stat=ierror )
ALLOCATE( RNGPROF(NPROF), stat=ierror )
if( ierror .ne. 0 ) then
  write(*,*)'**********ERROR IN RNGPROF ALLOCATION***********'
  stop
end if
RNGPROF = 0.

do i = 1, nprof
  read( 14, * ) rngprof(i)          !Range of profile in m
  do j = 0, lvlp-1
   read( 14, * ) hmsl(j,i), refmsl(j,i) !Height/refractivity levels
  end do
end do

!********************************READ TERRAIN INFO******************************

read( 14, * ) itp      !Number of terrain range/height points
read( 14, * ) igr      !Number of ground composition types

if( igr .gt. 0 ) then

  IF( ALLOCATED( DIELEC ) ) DEALLOCATE( DIELEC, stat=ierror )
  ALLOCATE( DIELEC(2, IGR), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN DIELEC ALLOCATION***********'
   stop
  end if
  DIELEC = 0.

  IF( ALLOCATED( IGRND ) ) DEALLOCATE( IGRND, stat=ierror )
  ALLOCATE( IGRND(IGR), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN IGRND ALLOCATION***********'
   stop
  end if
  IGRND = 0.
```

```
  IF( ALLOCATED( RGRND ) ) DEALLOCATE( RGRND, stat=ierror )
  ALLOCATE( RGRND(IGR), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN RGRND ALLOCATION***********'
   stop
  end if
  RGRND = 0.

! Read ranges at which ground types apply, ground composition types, and
!dielectric constants. If IGRND(i) = 7, then must specify non-zero values
!for DIELEC(), otherwise set to 0.

  do i = 1, igr
   read( 14, * ) rgrnd(i), igrnd(i), (dielec(j,i),j=1,2)
  end do

end if

if( itp .gt. 1 ) then ! Valid terrain profile must contain at least two
           ! height/range points.

  IF( ALLOCATED( TERX ) ) DEALLOCATE( TERX, stat=ierror )
  ALLOCATE( TERX(ITP), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN TERX ALLOCATION***********'
   stop
  end if
  TERX = 0.

  IF( ALLOCATED( TERY ) ) DEALLOCATE( TERY, stat=ierror )
  ALLOCATE( TERY(ITP), stat=ierror )
  if( ierror .ne. 0 ) then
   write(*,*)'**********ERROR IN TERY ALLOCATION***********'
   stop
  end if
  TERY = 0.

  do i = 1, itp
   read( 14, * ) terx(i), tery(i)
  end do
```

```
end if

close(14)

! Write all inputs that create the resulting output propagation loss values as
! part of log file.

ip = index( filein, '.' )
if( ip .gt. 0 ) then
  fileout = filein(1:ip-1)//'.out'
else
  ic = len_trim( filein )
  fileout = filein(1:ic)//'.out'
end if

open( 15, file=fileout )

write( 15, * )'****Input Log****'
write( 15, * )'lerr6 = ', lerr6
write( 15, * )'lerr12 = ', lerr12
write( 15, * )'Frequency (MHz) = ', freq
write( 15, * )'Antenna height (m) = ', antht
write( 15, * )'Antenna type = ', ipat
write( 15, * )'Polarization = ', ipol
write( 15, * )'Beamwidth (deg) = ', bwidth
write( 15, * )'Elevation angle (deg) = ', elev
write( 15, * )'Number of cut-back angles and factors = ', nfacs
if( nfacs .gt. 0 ) then
  write( 15, * )'Cut-back angles (deg) = ', ( hfang(i), i=1, nfacs )
  write( 15, * )'Cut-back factors = ', ( hffac(i), i=1, nfacs )
end if
write( 15, * )'Minimum output height (m) = ', hmin
write( 15, * )'Maximum output height (m) = ', hmax
write( 15, * )'Maximum output range (m) = ', rmax
write( 15, * )'Number of output height points = ', nzout
write( 15, * )'Number of output range points = ', nrout
write( 15, * )'Troposcatter flag = ', itropo
write( 15, * )'Number of refractivity profiles = ', nprof
write( 15, * )'Number of levels in refractivity profiles = ', lvlp
do j = 1, nprof
  write( 15, '(a,i2,a,f10.1)' )'Range of profile ', j, ' in m = ',rngprof(j)
```

```
  write(15,*)'Height (m)', '  M-unit for Profile', j
  do i = 0, lvlp-1
   write(15,*) hmsl(i,j), refmsl(i,j)
  end do
end do

write( 15, * )'Number of ground composition types = ', igr
write( 15, * )'Range(m) of ground type    Ground types                Dielec(perm.,cond.)'
do i = 1, igr
  write( 15, '(f15.2,20x,i1,7x,2(f15.2))' ) rgrnd(i), igrnd(i), (dielec(j,i), j=1,2)
end do

write( 15, * )'Number of terrain range/height points = ', itp
if( itp .gt. 1 ) then
  write(15,*)'Range (km)', 'Height (m)'
  do i = 1, itp
   write( 15, * ) terx(i)*1.e-3, tery(i)
  end do
end if

write( 15, * )
write( 15, * )'********Output Loss Values*******'

! Variables in CAPS are returned.

call apminit( IXOSTP, IERROR )

if( ierror .ne. 0 ) then
  write(*,*)'*********** ERROR IN APMINIT **************'
  write(*,*)'******** IERROR = ', ierror,' **********'
  stop
end if

do istp = 1, nrout

  call apmstep( istp, ROUT, MLOSS, JSTART, JEND )

  write(*,*)'range in km = ', rout*1.e-3 !Output to screen

! JSTART = start of valid loss points, JEND = end of valid loss
! points. If at a range where extended optics will be applied, then
```

```
! JEND will be the index at top of PE region in MLOSS().

! Store loss points in 2-dim. grid for later output to file.

  do m = jstart, jend
   itloss( istp, m ) = mloss(m)
  end do
end do

call xoinit( ixostp, jend, JXSTART, IERROR ) ! Initialize variables to be used
                       ! in XO model
if( ierror .gt. 0 ) then
  write(*,*)'*******ERROR IN XOINIT*******'
  stop
end if

! If extended optics model needs to be used, then call.

if( ixostp .gt. 0 ) then

  do istp = ixostp, nrout

   call xostep( istp, ROUT, MLOSS, jxstart, JXEND )

   write(*,*)'range in km (XO region) = ', rout*1.e-3 !Output to screen

   do m = jxstart, jxend
     itloss( istp, m ) = mloss(m)
   end do
  end do

end if

! Now store all loss values in output file FILEOUT.
! Recall that MLOSS is the propagation loss in centibels, i.e.,
! MLOSS() = NINT( propagation loss in dB * 10. ).

dzo = (hmax-hmin) / float( nzout ) !Determine height increment of
                 !output points.
dro = rmax / float( nrout )    !Determine range increment of output
                 !points.
```

```
do j = 1, nrout
  write(15,*)
  write(15,*)'range in km = ', float(j)*dro*1.e-3
  write(15,*)
  write(15,*)'Height (m)    Loss (dB)'
  do k = 1, nzout
   write(15,*) hmin + float(k)*dzo, itloss(j,k)*.1
  end do
end do
close(15)

! Call APMCLEAN to deallocate all allocated arrays used within APM routines.

call apmclean( IERROR )
if( ierror .gt. 0 ) then
  write(*,*)'*******ERROR IN APMCLEAN*******'
  stop
end if

!Deallocate all allocated arrays in main driver program (except IGRND(),
!RGRND(), and DIELEC() - this is done in APMCLEAN) before exiting.

deallocate( hmsl, refmsl, rngprof, itloss, mloss )
if( itp .gt. 1 ) deallocate( terx, tery )
if( nfacs .gt. 0 ) deallocate( hfang, hffac )

write(*,'(a\)')' Input another file? (y or n)'
read(*, '(a)' ) answer
if(( answer .eq. 'y' ) .or. ( answer .eq. 'Y')) goto 10

end!
```

8. INPUT FILE LISTINGS FOR TEST CASES

Each test case, when using the sample driver program APMMAIN.F90, shall consist of an input file (TestName.IN) and an output file (TestName.OUT). The input file's contents are listed in sections 8.1 through 8.28. The output file's contents, consisting of couplets of height in meters and propagation loss in dB, are listed in tables 16 through 43.

8.1 ABSORB.IN

```
.true. : LERR6 error flag
.true. : LERR12 error flag
20000. : Frequency in MHz
25.   : Antenna height in m
1     : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0     : Polarization (0=HOR, 1=VER)
5.    : Beamwidth in deg (this value is ignored for OMNI antenna)
0.    : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0     : Number of cut-back angles & factors (for specific ht-finder antenna)
0.    : Minimum output height in m
200.  : Maximum output height in m
50000. : Maximum output range in m
20    : Number of output height points
1     : Number of output range points
0     : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1     : Number of refractivity profiles
2     : Number of levels in refractivity profiles
0     : Extrapolation flag
0.    : Surface absolute humidity in g/m3
0.    : Surface air temperature in degrees Celsius
.146  : Gaseous absorption attenuation rate in dB/km
0.    : Range of first refractivity profiles in m
0.   350.    : Height & M-unit value of ref. profile 1, level 1
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0     : Number of terrain range/height points
1     : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.2 BLOCK.IN

```
.true. : LERR6 error flag
.true. : LERR12 error flag
1000.  : Frequency in MHz
25.   : Antenna height in m
1     : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0     : Polarization (0=HOR, 1=VER)
1.    : Beamwidth in deg (this value is ignored for OMNI antenna)
0.    : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0     : Number of cut-back angles & factors (for specific ht-finder antenna)
0.    : Minimum output height in m
1000.  : Maximum output height in m
50000. : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.    350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
6      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
0.     0.  : Range & height of terrain point 1
22500.  0.  : Range & height of terrain point 2
22500.  200. : Range & height of terrain point 3
27500.  200. : Range & height of terrain point 4
27500.  0.  : Range & height of terrain point 5
50000.  0.  : Range & height of terrain point 6
```

8.3 COSEC2.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
4       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.    350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.4 EDUCT.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
10000.  : Frequency in MHz
15.     : Antenna height in m
```

```
2       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
5.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
200.    : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
21      : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.     339.    : Height & M-unit value of ref. profile 1, level 1
.040    335.10  : Height & M-unit value of ref. profile 1, level 2
.1     333.66  : Height & M-unit value of ref. profile 1, level 3
.2     332.6   : Height & M-unit value of ref. profile 1, level 4
.398    331.54  : Height & M-unit value of ref. profile 1, level 5
.794    330.51  : Height & M-unit value of ref. profile 1, level 6
1.585   329.53  : Height & M-unit value of ref. profile 1, level 7
3.162   328.65  : Height & M-unit value of ref. profile 1, level 8
6.310   327.96  : Height & M-unit value of ref. profile 1, level 9
12.589  327.68  : Height & M-unit value of ref. profile 1, level 10
14.    327.67  : Height & M-unit value of ref. profile 1, level 11
25.119  328.13  : Height & M-unit value of ref. profile 1, level 12
39.811  329.25  : Height & M-unit value of ref. profile 1, level 13
50.119  330.18  : Height & M-unit value of ref. profile 1, level 14
63.096  331.44  : Height & M-unit value of ref. profile 1, level 15
79.433  333.12  : Height & M-unit value of ref. profile 1, level 16
100.    335.33  : Height & M-unit value of ref. profile 1, level 17
125.893 338.2   : Height & M-unit value of ref. profile 1, level 18
158.489 341.92  : Height & M-unit value of ref. profile 1, level 19
```

```
199.526 346.69  : Height & M-unit value of ref. profile 1, level 20
209.526 347.87  : Height & M-unit value of ref. profile 1, level 21
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.5 GASABS.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
20000.  : Frequency in MHz
25.    : Antenna height in m
1      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
5.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0      : Number of cut-back angles & factors (for specific height-finder antenna)
0.     : Minimum output height in m
200.    : Maximum output height in m
50000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0      : Extrapolation flag
10.     : Surface absolute humidity in g/m3
25.     : Surface air temperature in degrees Celsius
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.   350.   : Height & M-unit value of ref. profile 1, level 1
1000.  468.   : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.6 GAUSS.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.    : Antenna height in m
2      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
1.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0      : Number of cut-back angles & factors (for specific height-finder antenna)
0.     : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.7 HIBW.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.    : Antenna height in m
```

```
2       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
45.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.    350.    : Height & M-unit value of ref. profile 1, level 1
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.8 HIEL.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
2       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
10.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
20000.  : Maximum output height in m
```

```
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.9 HIFREQ.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
20000.  : Frequency in MHz
25.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
0.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
200.    : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
```

```
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.      350.     : Height & M-unit value of ref. profile 1, level 1
1000.   468.     : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.10 HITRAN.IN

```
.true.   : LERR6 error flag
.true.   : LERR12 error flag
1000.    : Frequency in MHz
100.    : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
0.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
1000.    : Maximum output height in m
50000.   : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.      350.     : Height & M-unit value of ref. profile 1, level 1
1000.   468.     : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
```

```
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.11 HORZ.IN

```
.true.   : LERR6 error flag
.true.   : LERR12 error flag
1000.    : Frequency in MHz
25.      : Antenna height in m
1        : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0        : Polarization (0=HOR, 1=VER)
1.       : Beamwidth in deg (this value is ignored for OMNI antenna)
0.       : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0        : Number of cut-back angles & factors (for specific height-finder antenna)
0.       : Minimum output height in m
2000.    : Maximum output height in m
50000.   : Maximum output range in m
20       : Number of output height points
1        : Number of output range points
0        : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1        : Number of refractivity profiles
2        : Number of levels in refractivity profiles
0        : Extrapolation flag
0.       : Surface absolute humidity in g/m3
0.       : Surface air temperature in degrees
0.       : Gaseous absorption attenuation rate in dB/km
0.       : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
0        : Number of terrain range/height points
1        : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.12 HTFIND.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.    : Antenna height in m
5      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
2.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0      : Number of cut-back angles & factors (for specific height-finder antenna)
0.     : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.13 LOBW.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.    : Antenna height in m
```

```
2       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
.5      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.    350.    : Height & M-unit value of ref. profile 1, level 1
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.14 LOEL.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
2       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
-10.    : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
20000.  : Maximum output height in m
```

```
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.      350.    : Height & M-unit value of ref. profile 1, level 1
1000.   468.    : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.15 LOFREQ.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
100.    : Frequency in MHz
25.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
0.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
5000.   : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
```

```
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.16 LOTRAN.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
1.     : Antenna height in m
1      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
0.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0      : Number of cut-back angles & factors (for specific height-finder antenna)
0.     : Minimum output height in m
10000.  : Maximum output height in m
50000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0·     : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
```

```
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.17 RDLONGB.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
150.   : Frequency in MHz
100.   : Antenna height in m
1      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
1.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0      : Number of cut-back angles & factors (for specific height-finder antenna)
0.     : Minimum output height in m
1000.   : Maximum output height in m
100000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
2      : Number of refractivity profiles
4      : Number of levels in refractivity profiles
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.   350.  : Height & M-unit value of ref. profile 1, level 1
0.   350.  : Height & M-unit value of ref. profile 1, level 2
0.   350.  : Height & M-unit value of ref. profile 1, level 3
1000. 468.  : Height & M-unit value of ref. profile 1, level 4
100000.  : Range of second refractivity profiles in m
0.   339.  : Height & M-unit value of ref. profile 2, level 1
250.  368.5 : Height & M-unit value of ref. profile 2, level 2
```

```
300.  319.  : Height & M-unit value of ref. profile 2, level 3
1000. 401.6 : Height & M-unit value of ref. profile 2, level 4
167     : Number of terrain range/height points
6     : Number of ground composition types
0., 2, 0., 0.     : Range, ground type (integer), permittivity, conductivity
28500., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
64800., 3, 0., 0.  : Range, ground type (integer), permittivity, conductivity
68700., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
74100., 4, 0., 0.  : Range, ground type (integer), permittivity, conductivity
100200., 0, 0., 0. : Range, ground type (integer), permittivity, conductivity
 0000.   8    : Range & height of terrain point 1 in meters
 0300.   8
 0600.   9
 0900.   9
 1200.   10
 1500.   11
 1800.   12
 2100.   13
 2400.   14
 2700.   15    : Range & height of terrain point 10 in meters
 3000.   17
 3300.   19
 3600.   21
 3900.   23
 4200.   25
 4500.   27
 4800.   28
 5100.   30
 5400.   31
 5700.   31    : Range & height of terrain point 20 in meters
 6000.   29
 6300.   23
 6600.   14
 6900.   9
 7200.   7
 7500.   7
 7800.   9
```

```
 8100.   11
 8400.   14
 8700.   13   : Range & height of terrain point 30 in meters
 9300.   13
 9600.   12
 9900.   11
10200.   8
10800.   8
11100.   7
12600.   7
12900.   6
14400.   6
14700.   7   : Range & height of terrain point 40 in meters
15000.   8
15300.   8
15600.   9
15900.   10
16200.   11
16500.   11
16800.   12
17400.   12
17700.   13
18000.   13   : Range & height of terrain point 50 in meters
18300.   14
18600.   15
18900.   16
19200.   18
19500.   20
19800.   21
20100.   22
20400.   23
20700.   24
21000.   24   : Range & height of terrain point 60 in meters
21300.   25
21600.   26
21900.   27
22200.   27
```

```
22500.   28
22800.   29
23400.   29
23700.   30
24600.   30
24900.   32   : Range & height of terrain point 70 in meters
25200.   34
25500.   38
26100.   38
26400.   36
26700.   34
27000.   32
27300.   27
27600.   15
27900.   6
28200.   1    : Range & height of terrain point 80 in meters
28500.   0
64500.   0
64800.   8
65100.   30
65400.   39
65700.   61
66600.   61
66900.   24
67200.   14
67500.   26   : Range & height of terrain point 90 in meters
67800.   16
68100.   1
68400.   1
68700.   0
73800.   0
74100.   1
74400.   1
74700.   10
75000.   8
75300.   39   : Range & height of terrain point 100 in meters
75600.   45
```

```
75900.   53
76200.   61
76500.   61
76800.   82
77100.   61
77400.   78
77700.   61
78000.  129
78300.   30   : Range & height of terrain point 110 in meters
78600.   46
78900.  159
79200.  184
79500.  226
79800.  152
80100.  201
80400.  244
80700.  152
81000.  143
81300.   91   : Range & height of terrain point 120 in meters
81600.  107
81900.  152
82200.  152
82500.  170
82800.  152
83100.   66
83400.   70
83700.  121
84000.  152
84300.  170   : Range & height of terrain point 130 in meters
84600.  141
84900.  139
85200.  147
85500.  177
85800.  152
86100.   61
86700.   61
87000.   70
```

```
87300.   44
87600.   11    : Range & height of terrain point 140 in meters
87900.   1
89400.   1
89700.   61
90000.   84
90300.  152
90600.  152
90900.  101
91200.   40
91500.   15
91800.   20    : Range & height of terrain point 150 in meters
92100.   2
92400.   10
92700.   4
93000.   1
93300.   1
93600.   0
93900.   1
96300.   1
96600.   0
96900.   1    : Range & height of terrain point 160 in meters
97500.   1
97800.   2
98100.   3
99300.   3
99600.   2
99900.   2
100200.   1          : Range & height of terrain point 167 in meters
```

8.18 RNGDEP.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
3000.   : Frequency in MHz
25.    : Antenna height in m
```

```
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
5.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
2000.     : Maximum output height in m
250000.   : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
2       : Number of refractivity profiles
4       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees Celsius
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.  330.   : Height & M-unit value of ref. profile 1, level 1
100. 342.5   : Height & M-unit value of ref. profile 1, level 2
230. 312.5   : Height & M-unit value of ref. profile 1, level 3
2000. 517.8   : Height & M-unit value of ref. profile 1, level 4
250000.   : Range of second refractivity profiles in m
0.  330.   : Height & M-unit value of ref. profile 2, level 1
600. 405.   : Height & M-unit value of ref. profile 2, level 2
730. 375.   : Height & M-unit value of ref. profile 2, level 3
2000. 522.3   : Height & M-unit value of ref. profile 2, level 4
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.   : Range, ground type (integer), permittivity, conductivity
```

8.19 SBDUCT.IN

```
.true.   : EF.LERR6 error flag
.true.   : EF.LERR12 error flag
3000.    : Frequency in MHz
25.     : Antenna height in m
```

```
2       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
5.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles and factors (for specific height-finder an-
tenna)
0.      : Minimum output height in m
5000.    : Maximum output height in m
200000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
4       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.    339.0     : Height & M-unit value of ref. profile 1, level 1
250.  368.5     : Height & M-unit value of ref. profile 1, level 2
300.  319.0     : Height & M-unit value of ref. profile 1, level 3
1000.  401.6    : Height & M-unit value of ref. profile 1, level 4
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.20 SINEX.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
3       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
```

```
0.     : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.     350.    : Height & M-unit value of ref. profile 1, level 1
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.21 TROPOS.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
100.   : Frequency in MHz
25.    : Antenna height in m
1      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
0.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0      : Number of cut-back angles & factors (for specific height-finder antenna)
0.     : Minimum output height in m
2000.   : Maximum output height in m
200000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
1      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
```

```
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.   350.   : Height & M-unit value of ref. profile 1, level 1
1000.  468.   : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.22 TROPOT.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
100.    : Frequency in MHz
25.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
0.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
2000.   : Maximum output height in m
200000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
1       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
```

```
0.     : Range of first refractivity profiles in m
0.   350.   : Height & M-unit value of ref. profile 1, level 1
1000.  468.   : Height & M-unit value of ref. profile 1, level 2
169    : Number of terrain range/height points
6      : Number of ground composition types
0., 2, 0., 0.  : Range, ground type (integer), permittivity, conductivity
28500., 0, 0., 0.
64800., 3, 0., 0.
68700., 0, 0., 0.
74100., 4, 0., 0.
100200., 0, 0., 0.
 0000.   8   : Range & height of terrain point 1 in meters
 0300.   8
 0600.   9
 0900.   9
 1200.   10
 1500.   11
 1800.   12
 2100.   13
 2400.   14
 2700.   15   : Range & height of terrain point 10 in meters
 3000.   17
 3300.   19
 3600.   21
 3900.   23
 4200.   25
 4500.   27
 4800.   28
 5100.   30
 5400.   31
 5700.   31   : Range & height of terrain point 20 in meters
 6000.   29
 6300.   23
 6600.   14
 6900.   9
 7200.   7
 7500.   7
```

```
 7800.   9
 8100.   11
 8400.   14
 8700.   13   : Range & height of terrain point 30 in meters
 9300.   13
 9600.   12
 9900.   11
10200.   8
10800.   8
11100.   7
12600.   7
12900.   6
14400.   6
14700.   7   : Range & height of terrain point 40 in meters
15000.   8
15300.   8
15600.   9
15900.   10
16200.   11
16500.   11
16800.   12
17400.   12
17700.   13
18000.   13   : Range & height of terrain point 50 in meters
18300.   14
18600.   15
18900.   16
19200.   18
19500.   20
19800.   21
20100.   22
20400.   23
20700.   24
21000.   24   : Range & height of terrain point 60 in meters
21300.   25
21600.   26
21900.   27
```

```
22200.   27
22500.   28
22800.   29
23400.   29
23700.   30
24600.   30
24900.   32   : Range & height of terrain point 70 in meters
25200.   34
25500.   38
26100.   38
26400.   36
26700.   34
27000.   32
27300.   27
27600.   15
27900.   6
28200.   1    : Range & height of terrain point 80 in meters
28500.   0
64500.   0
64800.   8
65100.   30
65400.   39
65700.   61
66600.   61
66900.   24
67200.   14
67500.   26   : Range & height of terrain point 90 in meters
67800.   16
68100.   1
68400.   1
68700.   0
73800.   0
74100.   1
74400.   1
74700.   10
75000.   8
75300.   39   : Range & height of terrain point 100 in meters
```

```
75600.   45
75900.   53
76200.   61
76500.   61
76800.   82
77100.   61
77400.   78
77700.   61
78000.  129
78300.   30    : Range & height of terrain point 110 in meters
78600.   46
78900.  159
79200.  184
79500.  226
79800.  152
80100.  201
80400.  244
80700.  152
81000.  143
81300.   91    : Range & height of terrain point 120 in meters
81600.  107
81900.  152
82200.  152
82500.  170
82800.  152
83100.   66
83400.   70
83700.  121
84000.  152
84300.  170    : Range & height of terrain point 130 in meters
84600.  141
84900.  139
85200.  147
85500.  177
85800.  152
86100.   61
86700.   61
```

```
87000.   70
87300.   44
87600.   11    : Range & height of terrain point 140 in meters
87900.   1
89400.   1
89700.   61
90000.   84
90300.  152
90600.  152
90900.  101
91200.   40
91500.   15
91800.   20    : Range & height of terrain point 150 in meters
92100.   2
92400.   10
92700.   4
93000.   1
93300.   1
93600.   0
93900.   1
96300.   1
96600.   0
96900.   1    : Range & height of terrain point 160 in meters
97500.   1
97800.   2
98100.   3
99300.   3
99600.   2
99900.   2
100200.   1
100200.   0.
200000.   0.   : Range & height of terrain point 167 in meters
```

8.23 USERHF.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.    : Antenna height in m
6      : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0      : Polarization (0=HOR, 1=VER)
1.     : Beamwidth in deg (this value is ignored for OMNI antenna)
0.     : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
10     : Number of cut-back angles & factors (for specific height-finder antenna)
1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5  : Cut-back angles in degrees
0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0  : Cut-back power factors
0.     : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20     : Number of output height points
1      : Number of output range points
0      : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1      : Number of refractivity profiles
2      : Number of levels in refractivity profiles
0      : Extrapolation flag
0.     : Surface absolute humidity in g/m3
0.     : Surface air temperature in degrees
0.     : Gaseous absorption attenuation rate in dB/km
0.     : Range of first refractivity profiles in m
0.    350.    : Height & M-unit value of ref. profile 1, level 1
1000.  468.    : Height & M-unit value of ref. profile 1, level 2
0      : Number of terrain range/height points
1      : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.24 VERT.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
```

```
1000.   : Frequency in MHz
25.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
1       : Polarization (0=HOR, 1=VER)
0.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
2000.   : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.    350.    : Height & M-unit value of ref. profile 1, level 1
1000.  468.   : Height & M-unit value of ref. profile 1, level 2
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.25 VERTMIX.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
100.    : Frequency in MHz
10.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
1       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
```

```
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
1000.    : Maximum output height in m
50000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.     350.     : Height & M-unit value of ref. profile 1, level 1
1000.  468.     : Height & M-unit value of ref. profile 1, level 2
2       : Number of terrain range/height points
2       : Number of ground composition types
0., 4, 0., 0.     : Range, ground type (integer), permittivity, conductivity
25000., 0, 0., 0.   : Range, ground type (integer), permittivity, conductivity
0.     0.  : Range & height of terrain point 1
50000.  0.  : Range & height of terrain point 2
```

8.26 VERTSEA.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
100.    : Frequency in MHz
25.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
1       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
1000.    : Maximum output height in m
```

```
300000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
4       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.   339.0    : Height & M-unit value of ref. profile 1, level 1
250.  368.5    : Height & M-unit value of ref. profile 1, level 2
300.  319.0    : Height & M-unit value of ref. profile 1, level 3
1000.  401.6    : Height & M-unit value of ref. profile 1, level 4
0       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
```

8.27 VERTUSRD.IN

```
.true.   : LERR6 error flag
.true.   : LERR12 error flag
100.    : Frequency in MHz
10.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
1       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
1000.    : Maximum output height in m
50000.   : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
```

```
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.      350.    : Height & M-unit value of ref. profile 1, level 1
1000.   468.    : Height & M-unit value of ref. profile 1, level 2
2       : Number of terrain range/height points
1       : Number of ground composition types
0., 7, 3., 6.e-4  : Range, ground type (integer), permittivity, conductivity
0.      0.  : Range & height of terrain point 1
50000.  0.  : Range & height of terrain point 2
```

8.28 WEDGE.IN

```
.true.  : LERR6 error flag
.true.  : LERR12 error flag
1000.   : Frequency in MHz
25.     : Antenna height in m
1       : Antenna type (1=OMNI,2=GAUSS,3=SINC(X),4=COSEC2,5=HTFIND,6=USRHTFIND)
0       : Polarization (0=HOR, 1=VER)
1.      : Beamwidth in deg (this value is ignored for OMNI antenna)
0.      : Antenna elevation angle in deg (this value is ignored for OMNI antenna)
0       : Number of cut-back angles & factors (for specific height-finder antenna)
0.      : Minimum output height in m
1000.   : Maximum output height in m
100000.  : Maximum output range in m
20      : Number of output height points
1       : Number of output range points
0       : Troposcatter flag: 0=no troposcatter, 1=troposcatter
1       : Number of refractivity profiles
2       : Number of levels in refractivity profiles
0       : Extrapolation flag
0.      : Surface absolute humidity in g/m3
0.      : Surface air temperature in degrees
```

```
0.      : Gaseous absorption attenuation rate in dB/km
0.      : Range of first refractivity profiles in m
0.      350.     : Height & M-unit value of ref. profile 1, level 1
1000.   468.     : Height & M-unit value of ref. profile 1, level 2
5       : Number of terrain range/height points
1       : Number of ground composition types
0., 0, 0., 0.  : Range, ground type (integer), permittivity, conductivity
0.        0.  : Range & height of terrain point 1
45000.      0.  : Range & height of terrain point 2
50000.      200. : Range & height of terrain point 3
55000.      0.  : Range & height of terrain point 4
100000.     0.  : Range & height of terrain point 5
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1998	Final

4. TITLE AND SUBTITLE

ADVANCED PROPAGATION MODEL (APM)
COMPUTER SOFTWARE CONFIGURATION ITEM (CSCI)
DOCUMENTS

5. FUNDING NUMBERS

PE: 0602173C
AN: DN306062
WU: D88–MP67

6. AUTHOR(S)

D. B. Sailors, A. E. Barrios, W. L. Patterson, H. V. Hitney

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Space and Naval Warfare Systems Center
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Space and Naval Warfare Systems Command
PMW–185
53660 Oceanview Drive
San Diego, CA 92152–5002

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

TD 3033

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document describes the Advanced Propagation Mode (APM) Version 1.0 computer software configuration item (CSCI) design and provides an input software requirement overview, a CSCI design architecture overview, and a detailed design description of each CSCI component.

14. SUBJECT TERMS

Mission Area: Command, Control, and Communications
electromagnetic propagation
tactical fleet applications
software requirements

15. NUMBER OF PAGES

428

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
A. E. Barrios	(619) 553–1429 e–mail: barrios@spawar.navy.mil	Code D883

INITIAL DISTRIBUTION

Code D0012	Patent Counsel	(1)
Code D0271	Archive/Stock	(6)
Code D0274	Library	(2)
Code D027	M. E. Cathcart	(1)
Code D0271	D. Richter	(1)
Code D88	J. Richter	(1)
Code D882	D. Sailors	(1)
Code D883	A. Barrios	(25)
Code D883	H. Hitney	(1)
Code D883	W. Patterson	(1)
Code D883	R. Paulus	(1)

Defense Technical Information Center
Fort Belvoir, VA 22060–6218 (4)

SPAWARSYSCEN Liaison Office
Arlington, VA 22202–4804

Center for Naval Analyses
Alexandria, VA 22302–0268

Navy Acquisition, Research and Development
Information Center (NARDIC)
Arlington, VA 22244–5114

GIDEP Operations Center
Corona, CA 91718–8000